In [1]:
```
#importing libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Import and Analyze the Dataset

Import the Dataset:

In [2]:
```
movies = pd.read_csv("netflix_title.csv")
movies.head(3)
```

Out[2]:

	show_id	type	title	director	cast	country	date_added	release_year	rati
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	T M
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	T M

Check the Structure and Characteristics:

- Display the first few rows: `df.head()`
- Summary of the dataset: `df.info()`
- Descriptive statistics: `df.describe()`

In [3]:
```
movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

In [4]:
```
movies.describe(include='all')
```

Out[4]:

	show_id	type	title	director	cast	country	date_added	release_yea
count	8807	8807	8807	6173	7982	7976	8797	8807.00000
unique	8807	2	8807	4528	7692	748	1767	Na
top	s1	Movie	Dick Johnson Is Dead	Rajiv Chilaka	David Attenborough	United States	January 1, 2020	Na
freq	1	6131	1	19	19	2818	109	Na
mean	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2014.18019
std	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8.81931
min	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1925.00000
25%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2013.00000
50%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2017.00000
75%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2019.00000
max	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2021.00000

Descriptive Statistics Report

Overview

- The dataset contains a list of TV shows and movies available on Netflix, consisting of 8807 records with the following attributes: show_id, type, title, director, cast, country,

date_added, release_year, rating, duration, listed_in, and description.

- Count and Uniqueness
  - show_id: 8807 unique values. Each show/movie has a unique identifier.
  - type: 2 unique values (Movie, TV Show). Majority are Movies (6131).
  - title: 8807 unique values. Each show/movie has a unique title.
  - director: 4528 unique values. There are 6173 entries with a director name.
  - cast: 7692 unique values. There are 7982 entries with cast information.
  - country: 748 unique values. There are 7976 entries with country information.
  - date_added: 1767 unique values. There are 8797 entries with the date added to Netflix.
  - release_year: Numerical, ranging from 1925 to 2021.
  - rating: 17 unique values. Most frequent rating is TV-MA (3207 entries).
  - duration: 220 unique values. Represented either in minutes (for movies) or number of seasons (for TV shows).
  - listed_in: 514 unique values. Categories or genres the content belongs to.
  - description: 8775 unique values. Short summaries of the shows/movies.

Detect Null Values and Outliers

In [5]:
```
#checking null values in every column of our data
movies.isna().sum()
```

Out[5]:
```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added        10
release_year       0
rating             4
duration           3
listed_in          0
description        0
dtype: int64
```

In [6]:
```
#number of unique values in our data
for i in movies.columns:
  print(i,':',movies[i].nunique())
```

```
show_id : 8807
type : 2
title : 8807
director : 4528
cast : 7692
country : 748
date_added : 1767
release_year : 74
rating : 17
duration : 220
listed_in : 514
description : 8775
```

In [7]:
```
# checking Nested Data in colums
print( np.any(movies["director"].str.contains(",")))
print(np.any(movies["listed_in"].str.contains(",")))
print(np.any(movies["cast"].str.contains(",")))
print(np.any(movies["director"].str.contains(",")))
```

```
True
True
True
True
```

In [8]:
```
#unnesting the cast column, i.e- creating separate lines for each cast in a movie
actors = movies["cast"].apply(lambda x : str(x).split(",")).to_list()
actors = pd.DataFrame(actors,index=movies["title"])
actors.head(3)
ok1 = actors.stack()
actors = pd.DataFrame(ok1.reset_index())
actors.drop({"level_1"}, axis=1,inplace=True)
actors.rename({0:"Actors"},axis=1,inplace=True)
actors.head(2)
```

Out[8]:

	title	Actors
0	Dick Johnson Is Dead	nan
1	Blood & Water	Ama Qamata

In [9]:
```
# unnesting the directors column, i.e- creating separate lines for each director in
director = movies["director"].apply(lambda x : str(x).split(",")).to_list()
director = pd.DataFrame(director,index=movies["title"])
director.head(3)
ok1 = director.stack()
director = pd.DataFrame(ok1.reset_index())
director.drop({"level_1"}, axis=1,inplace=True)
director.rename({0:"Director"},axis=1,inplace=True)
director.head(2)
```

Out[9]:

	title	Director
0	Dick Johnson Is Dead	Kirsten Johnson
1	Blood & Water	nan

In [10]:
```
#unnesting the listed_in column, i.e- creating separate lines for each genre in a m
listed_in = movies["listed_in"].apply(lambda x : str(x).split(",")).to_list()
listed_in = pd.DataFrame(listed_in,index=movies["title"])
listed_in.head(3)
ok1 = listed_in.stack()
listed_in = pd.DataFrame(ok1.reset_index())
listed_in.drop({"level_1"}, axis=1,inplace=True)
listed_in.rename({0:"Genre"},axis=1,inplace=True)
listed_in.head(2)
```

Out[10]:

	title	Genre
0	Dick Johnson Is Dead	Documentaries
1	Blood & Water	International TV Shows

In [11]:
```
#unnesting the country column, i.e- creating separate lines for each country in a m
country = movies["country"].apply(lambda x : str(x).split(",")).to_list()
country = pd.DataFrame(country,index=movies["title"])
country.head(3)
ok1 = country.stack()
country = pd.DataFrame(ok1.reset_index())
country.drop({"level_1"}, axis=1,inplace=True)
country.rename({0:"country"},axis=1,inplace=True)
country.head(2)
```

Out[11]:

	title	country
0	Dick Johnson Is Dead	United States
1	Blood & Water	South Africa

In [12]:
```
#merging the unnested director data with unnested actors data
ac_dr    = actors.merge(director,how="inner",on="title")
#merging the above merged data with unnested genre data
list_all =ac_dr.merge(listed_in,how="inner",on="title")
#merging the above merged data with unnested country data
all_con  =list_all.merge(country,how="inner",on="title")
all_con.head(4)
```

Out[12]:

	title	Actors	Director	Genre	country
0	Dick Johnson Is Dead	nan	Kirsten Johnson	Documentaries	United States
1	Blood & Water	Ama Qamata	nan	International TV Shows	South Africa
2	Blood & Water	Ama Qamata	nan	TV Dramas	South Africa
3	Blood & Water	Ama Qamata	nan	TV Mysteries	South Africa

In [13]:
```
#merging our unnested data with the original data
df_final = all_con.merge(movies[['show_id', 'type', 'title','date_added',
       'release_year', 'rating', 'duration']],how="inner",on="title")
df_final.head(5)
```

Out[13]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	nan	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	nan	International TV Shows	South Africa	s2	TV Show	September 24, 2021	
2	Blood & Water	Ama Qamata	nan	TV Dramas	South Africa	s2	TV Show	September 24, 2021	
3	Blood & Water	Ama Qamata	nan	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	
4	Blood & Water	Khosi Ngema	nan	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [14]:
```
#now checking nulls
df_final.isna().sum()
```

Out[14]:
```
title            0
Actors           0
Director         0
Genre            0
country          0
show_id          0
type             0
date_added     158
release_year     0
rating          67
duration         3
dtype: int64
```

In [15]:
```
df1 = df_final
```

In [16]:
```
#date added column is imputed on the basis of release year,i.e- suppose there's a n
#when release year was 2013.So below piece of code just checks the mode of date add
# and imputes in place of nulls the corresponding mode

for year in df1.loc[df1["date_added"].isna(),"release_year"].unique():
    mode = df1.loc[df1["release_year"]==year,"date_added"].mode().values[0]
    df1.loc[df1["release_year"]==year,"date_added"] = df1.loc[df1["release_year"]==
```

In [17]:
```
df1.isnull().sum()
```

Out[17]:
```
title           0
Actors          0
Director        0
Genre           0
country         0
show_id         0
type            0
date_added      0
release_year    0
rating         67
duration        3
dtype: int64
```

In [18]:
```
#checking the occurences of each of the ratings
movies['rating'].value_counts()
```

Out[18]:
```
rating
TV-MA       3207
TV-14       2160
TV-PG        863
R            799
PG-13        490
TV-Y7        334
TV-Y         307
PG           287
TV-G         220
NR            80
G             41
TV-Y7-FV       6
NC-17          3
UR             3
74 min         1
84 min         1
66 min         1
Name: count, dtype: int64
```

In duration column, it was observed that the nulls had values which were written in corresponding ratings column, i.e- you can't expect ratings to be in min. So the duration column nulls are replaced by corresponding values in ratings column

In [19]:
```
df1.loc[df1["duration"].isna(),'duration'] = df1.loc[df1["duration"].isna(),"rating
```

In [20]:
```
df1.isnull().sum()
```

Out[20]:
```
title           0
Actors          0
Director        0
Genre           0
country         0
show_id         0
type            0
date_added      0
release_year    0
rating         67
duration        0
dtype: int64
```

In [21]:
```
#Ratings can't be in min, so it has been made NR(i.e- Non Rated)
df1.loc[df1["rating"].str.contains("min",na=False),"rating"]="NR"
```

In [22]:
```
df1["rating"] = df1["rating"].fillna("NR")
```

In [23]:
```
df1.isna().sum()
```

Out[23]:
```
title           0
Actors          0
Director        0
Genre           0
country         0
show_id         0
type            0
date_added      0
release_year    0
rating          0
duration        0
dtype: int64
```

In [24]:
```
df2 = df1
```

replacing nan values of director and actor by Unknown Actor and Director

In [25]:
```
df2["Actors"] = df2["Actors"].replace(["nan"],["Unknow Actors"])
df2["Director"] = df2["Director"].replace(["nan"],["Unknow Director"])
df2["country"] = df2["country"].replace(["nan"],[np.nan])
df2.isnull().sum()
```

Out[25]:
```
title              0
Actors             0
Director           0
Genre              0
country        11897
show_id            0
type               0
date_added         0
release_year       0
rating             0
duration           0
dtype: int64
```

In [26]:
```
#just an attempt to observe nulls in country column
df2.loc[df2["country"].isnull()].head(4)
```

Out[26]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
58	Ganglands	Sami Bouajila	Julien Leclercq	Crime TV Shows	NaN	s3	TV Show	September 24, 2021	
59	Ganglands	Sami Bouajila	Julien Leclercq	International TV Shows	NaN	s3	TV Show	September 24, 2021	
60	Ganglands	Sami Bouajila	Julien Leclercq	TV Action & Adventure	NaN	s3	TV Show	September 24, 2021	
61	Ganglands	Tracy Gotoas	Julien Leclercq	Crime TV Shows	NaN	s3	TV Show	September 24, 2021	

In [27]:
```
#country column is imputed on the basis of director,i.e- suppose there's a null for
#when we have a director whose other movies have a country given.So below piece of
#country for the director
# and imputes in place of nulls the corresponding mode

for i  in df2.loc[df2["country"].isna(),"Director"].unique():
    if i in df2.loc[~df2["country"].isna(),"Director"].unique():
        imp = df2.loc[df2["Director"]==i,"country"].mode().values[0]
        df2.loc[df2["Director"]==i,"country"] = df2.loc[df2["Director"]==i,"country
```

So we imputed the country column on the basis of directors whose other movie titles had countries given. But there might be directors who have only one occurence in our data. In that scenario, I have used Actors as a basis. i.e- for this Actor majorly acts in movies of which country? Imputation has been done on this basis. For remaining rows, country has been filled as Unknown Country

In [28]:
```
for i  in df2.loc[df2["country"].isna(),"Actors"].unique():
    if i in df2.loc[~df2["country"].isna(),"Actors"].unique():
        imp = df2.loc[df2["Actors"]==i,"country"].mode().values[0]
        df2.loc[df2["Actors"]==i,"country"] = df2.loc[df2["Actors"]==i,"country"].f
```

In [29]:
```
df2.isna().sum()
```

Out[29]:
```
title              0
Actors             0
Director           0
Genre              0
country         2455
show_id            0
type               0
date_added         0
release_year       0
rating             0
duration           0
dtype: int64
```

If there are still nulls, I just replace it by Unknown Country

In [30]:
```
df2["country"] = df2["country"].fillna("Unknow Country")
```

In [31]:
```
#now checking nulls
df2.isna().sum()
```

Out[31]:
```
title           0
Actors          0
Director        0
Genre           0
country         0
show_id         0
type            0
date_added      0
release_year    0
rating          0
duration        0
dtype: int64
```

In [32]:
```
df2.head(5)
```

Out[32]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	
2	Blood & Water	Ama Qamata	Unknow Director	TV Dramas	South Africa	s2	TV Show	September 24, 2021	
3	Blood & Water	Ama Qamata	Unknow Director	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	
4	Blood & Water	Khosi Ngema	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [33]:
```
df_final=df2
```

In [34]:
```
df_final.head()
```

Out[34]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	
2	Blood & Water	Ama Qamata	Unknow Director	TV Dramas	South Africa	s2	TV Show	September 24, 2021	
3	Blood & Water	Ama Qamata	Unknow Director	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	
4	Blood & Water	Khosi Ngema	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [35]:
```
df_final["duration"].value_counts()
```

Out[35]:
```
duration
1 Season     35035
2 Seasons     9559
3 Seasons     5084
94 min        4343
106 min       4040
             ...
3 min            4
5 min            3
11 min           2
8 min            2
9 min            2
Name: count, Length: 220, dtype: int64
```

In [36]:
```
#removing mins from data
df_final["duration"]=df_final["duration"].str.replace("min","")
```

In [37]:
```
df_final1 = df_final
```

In [ ]:
```

```

In [38]:
```
df_final['duration'].unique()
```

Out[38]:
```
array(['90 ', '2 Seasons', '1 Season', '91 ', '125 ', '9 Seasons', '104 ',
       '127 ', '4 Seasons', '67 ', '94 ', '5 Seasons', '161 ', '61 ',
       '166 ', '147 ', '103 ', '97 ', '106 ', '111 ', '3 Seasons', '110 ',
       '105 ', '96 ', '124 ', '116 ', '98 ', '23 ', '115 ', '122 ', '99 ',
       '88 ', '100 ', '6 Seasons', '102 ', '93 ', '95 ', '85 ', '83 ',
       '113 ', '13 ', '182 ', '48 ', '145 ', '87 ', '92 ', '80 ', '117 ',
       '128 ', '119 ', '143 ', '114 ', '118 ', '108 ', '63 ', '121 ',
       '142 ', '154 ', '120 ', '82 ', '109 ', '101 ', '86 ', '229 ',
       '76 ', '89 ', '156 ', '112 ', '107 ', '129 ', '135 ', '136 ',
       '165 ', '150 ', '133 ', '70 ', '84 ', '140 ', '78 ', '7 Seasons',
       '64 ', '59 ', '139 ', '69 ', '148 ', '189 ', '141 ', '130 ',
       '138 ', '81 ', '132 ', '10 Seasons', '123 ', '65 ', '68 ', '66 ',
       '62 ', '74 ', '131 ', '39 ', '46 ', '38 ', '8 Seasons',
       '17 Seasons', '126 ', '155 ', '159 ', '137 ', '12 ', '273 ', '36 ',
       '34 ', '77 ', '60 ', '49 ', '58 ', '72 ', '204 ', '212 ', '25 ',
       '73 ', '29 ', '47 ', '32 ', '35 ', '71 ', '149 ', '33 ', '15 ',
       '54 ', '224 ', '162 ', '37 ', '75 ', '79 ', '55 ', '158 ', '164 ',
       '173 ', '181 ', '185 ', '21 ', '24 ', '51 ', '151 ', '42 ', '22 ',
       '134 ', '177 ', '13 Seasons', '52 ', '14 ', '53 ', '8 ', '57 ',
       '28 ', '50 ', '9 ', '26 ', '45 ', '171 ', '27 ', '44 ', '146 ',
       '20 ', '157 ', '17 ', '203 ', '41 ', '30 ', '194 ', '15 Seasons',
       '233 ', '237 ', '230 ', '195 ', '253 ', '152 ', '190 ', '160 ',
       '208 ', '180 ', '144 ', '5 ', '174 ', '170 ', '192 ', '209 ',
       '187 ', '172 ', '16 ', '186 ', '11 ', '193 ', '176 ', '56 ',
       '169 ', '40 ', '10 ', '3 ', '168 ', '312 ', '153 ', '214 ', '31 ',
       '163 ', '19 ', '12 Seasons', '179 ', '11 Seasons', '43 ', '200 ',
       '196 ', '167 ', '178 ', '228 ', '18 ', '205 ', '201 ', '191 '],
      dtype=object)
```

In [39]:
```
df_final["duration_copy"] = df_final["duration"].copy()
df_final.head(2)
```

Out[39]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [40]:
```
df_final.loc[df_final["duration_copy"].str.contains("Season"),"duration_copy"]=0
df_final.head()
```

Out[40]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	
2	Blood & Water	Ama Qamata	Unknow Director	TV Dramas	South Africa	s2	TV Show	September 24, 2021	
3	Blood & Water	Ama Qamata	Unknow Director	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	
4	Blood & Water	Khosi Ngema	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [41]:
```
df_final["duration_copy"] = df_final["duration_copy"].astype(int)
```

In [42]:
```
df_final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 202065 entries, 0 to 202064
Data columns (total 12 columns):
 #   Column         Non-Null Count   Dtype 
---  ------         --------------   ----- 
 0   title          202065 non-null  object
 1   Actors         202065 non-null  object
 2   Director       202065 non-null  object
 3   Genre          202065 non-null  object
 4   country        202065 non-null  object
 5   show_id        202065 non-null  object
 6   type           202065 non-null  object
 7   date_added     202065 non-null  object
 8   release_year   202065 non-null  int64 
 9   rating         202065 non-null  object
 10  duration       202065 non-null  object
 11  duration_copy  202065 non-null  int32 
dtypes: int32(1), int64(1), object(10)
memory usage: 17.7+ MB
```

In [43]:
```
df_final['duration_copy'].describe()
```

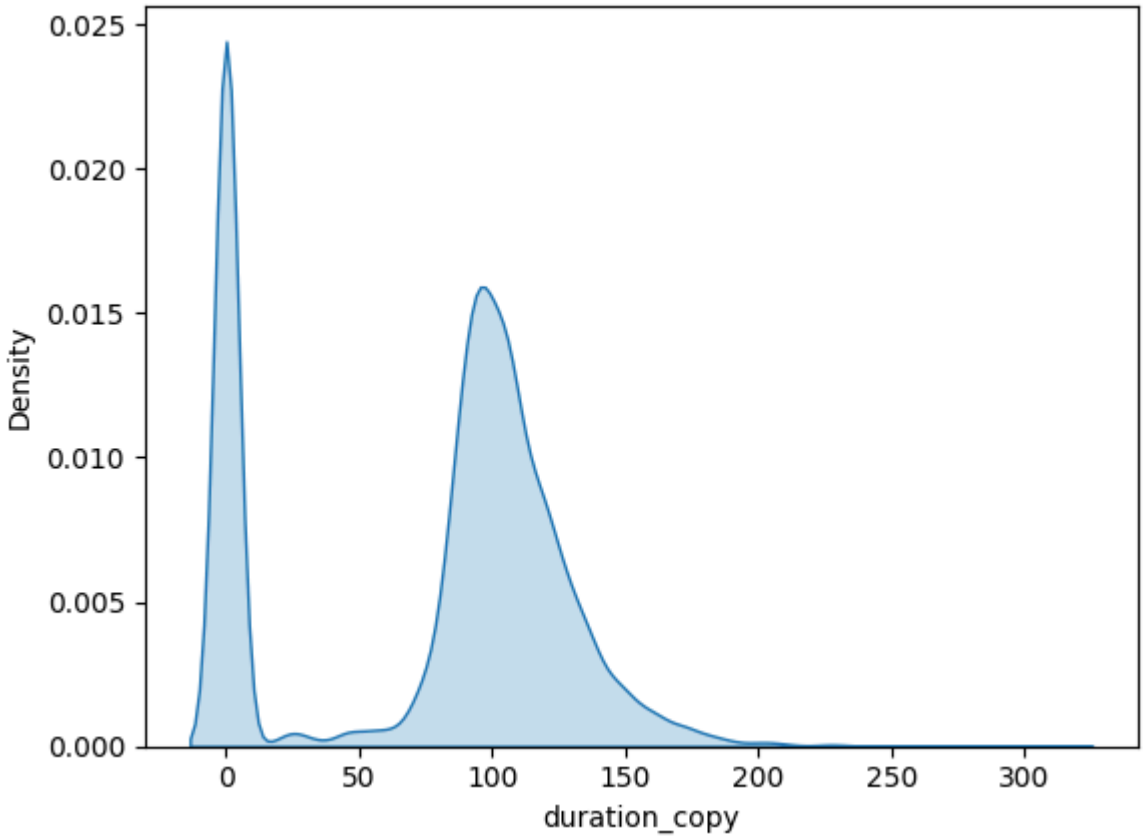
Out[43]:
```
count    202065.000000
mean         77.152065
std          52.262613
min           0.000000
25%           0.000000
50%          95.000000
75%         112.000000
max         312.000000
Name: duration_copy, dtype: float64
```

In [44]:
```
df_final.head(5)
```

Out[44]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	
2	Blood & Water	Ama Qamata	Unknow Director	TV Dramas	South Africa	s2	TV Show	September 24, 2021	
3	Blood & Water	Ama Qamata	Unknow Director	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	
4	Blood & Water	Khosi Ngema	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [45]:
```
sns.kdeplot(df_final["duration_copy"],fill=True)
plt.show()
```



In [46]:
```
# A/c to observation and Visulaiztion techinque we found Season is more number of c
```

In [47]:
```
# convert Duration coulmn into categorical data
bins = [-1,1,50,80,100,120,150,200,315]
lebal = ["<1","1-50","50-80","80-100","100-120","120-150","150-200","200-315"]
df_final["duration_copy"] = pd.cut(df_final["duration_copy"],bins=bins,labels=lebal
df_final.head(2)
```

Out[47]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [48]:
```
df_final.loc[~df_final["duration"].str.contains("Seasons"),"duration"] = df_final.l
df_final.head(2)
```

Out[48]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [49]:
```
df_final["duration"].unique()
```

Out[49]:
```
array(['80-100', '2 Seasons', '<1', '120-150', '9 Seasons', '100-120',
       '4 Seasons', '50-80', '5 Seasons', '150-200', '3 Seasons', '1-50',
       '6 Seasons', '200-315', '7 Seasons', '10 Seasons', '8 Seasons',
       '17 Seasons', '13 Seasons', '15 Seasons', '12 Seasons',
       '11 Seasons'], dtype=object)
```

In [50]:
```
df_final.drop("duration_copy",axis=1,inplace=True)
```

In [51]:
```
df_final.head(2)
```

Out[51]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [52]:
```
df_final3 = df_final
```

In [53]:
```
df_final3.head(3)
```

Out[53]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	
2	Blood & Water	Ama Qamata	Unknow Director	TV Dramas	South Africa	s2	TV Show	September 24, 2021	

In [54]:
```
# parse date_added columnn date into date format "y-m-d" then put values into new c
from datetime import datetime
from dateutil.parser import parse
arr = []
for i in df_final3["date_added"].values:
    dt1 = parse(i)
    arr.append(dt1.strftime("%Y-%m-%d"))
```

In [55]:
```
df_final3["modifiy-date-added"] = arr
```

In [56]:
```
df_final3.head(3)
```

Out[56]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	
2	Blood & Water	Ama Qamata	Unknow Director	TV Dramas	South Africa	s2	TV Show	September 24, 2021	

In [57]:
```
# convert modifiy-date-added column into datetime format
df_final3["modifiy-date-added"] = pd.to_datetime(df_final3["modifiy-date-added"])
```

In [58]:
```
# just attemp to observe change datatype of modifiy-date-added column
df_final3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 202065 entries, 0 to 202064
Data columns (total 12 columns):
 #   Column              Non-Null Count   Dtype         
---  ------              --------------   -----         
 0   title               202065 non-null  object        
 1   Actors              202065 non-null  object        
 2   Director            202065 non-null  object        
 3   Genre               202065 non-null  object        
 4   country             202065 non-null  object        
 5   show_id             202065 non-null  object        
 6   type                202065 non-null  object        
 7   date_added          202065 non-null  object        
 8   release_year        202065 non-null  int64         
 9   rating              202065 non-null  object        
 10  duration            202065 non-null  object        
 11  modifiy-date-added  202065 non-null  datetime64[ns]
dtypes: datetime64[ns](1), int64(1), object(10)
memory usage: 18.5+ MB
```

In [59]:
```
# extract year from modifiy-date-added column and put values into new column year
df_final3["year"] = df_final3["modifiy-date-added"].dt.year
df_final3.head(2)
```

Out[59]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [ ]:
```

```

In [60]:
```
# extract month from modifiy-date-added column and put values into new column month
df_final3["month"]=df_final3["modifiy-date-added"].dt.month
df_final3.head(2)
```

Out[60]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [61]:
```
# extract week from modifiy-date-added column and put values into new column week
df_final3["week"] = df_final3["modifiy-date-added"].dt.isocalendar().week
```

```
df_final3.head(2)
```

Out[61]:

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
0	Dick Johnson Is Dead	Unknow Actors	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknow Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	

In [62]:
```
#Titles such as Bahubali(Hindi Version),Bahubali(Tamil Version) were there. Since i
# observe is there any presence of brackets and content between brackets.
df_final3.loc[df_final3["title"].str.contains("\(.*\)")].head(5)
```

Out[62]:

	title	Actors	Director	Genre	country	show_id	type	date_adde
1833	Tughlaq Durbar (Telugu)	Vijay Sethupathi	Delhiprasad Deenadayalan	Comedies	India	s80	Movie	Septembe 11, 202
1834	Tughlaq Durbar (Telugu)	Vijay Sethupathi	Delhiprasad Deenadayalan	Dramas	India	s80	Movie	Septembe 11, 202
1835	Tughlaq Durbar (Telugu)	Vijay Sethupathi	Delhiprasad Deenadayalan	International Movies	India	s80	Movie	Septembe 11, 202
1836	Tughlaq Durbar (Telugu)	Parthiban	Delhiprasad Deenadayalan	Comedies	Unknow Country	s80	Movie	Septembe 11, 202
1837	Tughlaq Durbar (Telugu)	Parthiban	Delhiprasad Deenadayalan	Dramas	Unknow Country	s80	Movie	Septembe 11, 202

In [63]:
```
#presence of brackets and content between brackets is removed.
df_final3["title"] = df_final3["title"].str.replace("\(.*\)","",regex=True)
df_final.loc[1833:1837]
```

Out[63]:

	title	Actors	Director	Genre	country	show_id	type	date_adde
1833	Tughlaq Durbar	Vijay Sethupathi	Delhiprasad Deenadayalan	Comedies	India	s80	Movie	Septembe 11, 202
1834	Tughlaq Durbar	Vijay Sethupathi	Delhiprasad Deenadayalan	Dramas	India	s80	Movie	Septembe 11, 202
1835	Tughlaq Durbar	Vijay Sethupathi	Delhiprasad Deenadayalan	International Movies	India	s80	Movie	Septembe 11, 202
1836	Tughlaq Durbar	Parthiban	Delhiprasad Deenadayalan	Comedies	Unknow Country	s80	Movie	Septembe 11, 202
1837	Tughlaq Durbar	Parthiban	Delhiprasad Deenadayalan	Dramas	Unknow Country	s80	Movie	Septembe 11, 202

Univariate Analysis in terms of counts of each column

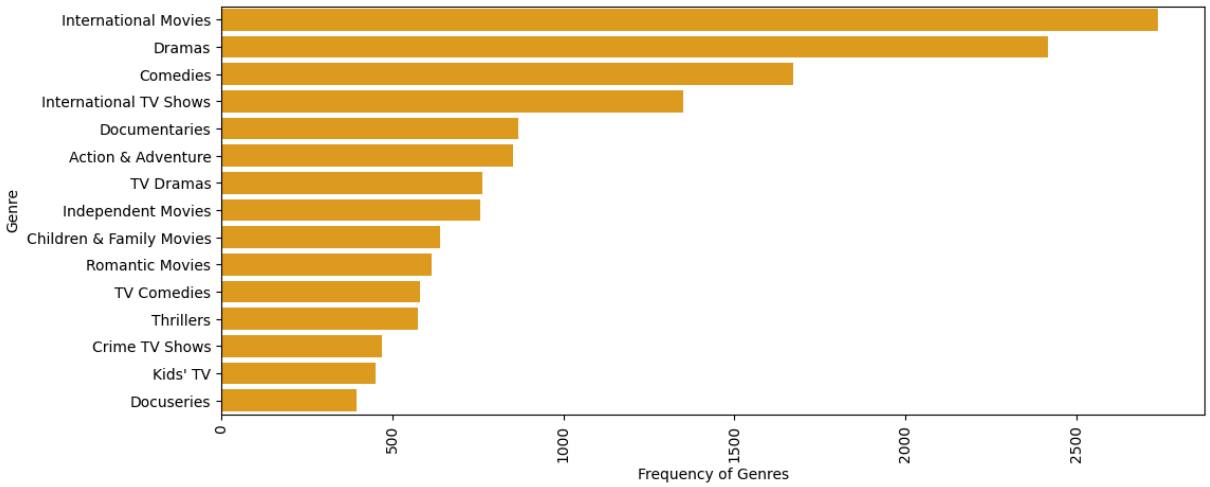
In [66]:
```
df_final3.loc[df_final3["Genre"].duplicated(),"Genre"].unique()
```

Out[66]:
```
array(['International TV Shows', ' TV Dramas', ' TV Mysteries',
       'Crime TV Shows', ' International TV Shows',
       ' TV Action & Adventure', ' Romantic TV Shows', ' TV Comedies',
       'TV Dramas', ' TV Horror', 'Children & Family Movies', 'Dramas',
       ' Independent Movies', ' International Movies', ' Reality TV',
       'British TV Shows', 'Comedies', ' Dramas', ' Comedies',
       ' Docuseries', 'TV Comedies', 'Documentaries',
       ' Spanish-Language TV Shows', 'Thrillers', ' Romantic Movies',
       'Docuseries', ' Music & Musicals', 'Horror Movies',
       ' Sci-Fi & Fantasy', ' TV Thrillers', "Kids' TV", ' Thrillers',
       'Action & Adventure', ' TV Sci-Fi & Fantasy', ' Classic Movies',
       ' Horror Movies', ' Anime Features', 'Reality TV',
       ' Sports Movies', 'Anime Series', " Kids' TV",
       'International Movies', ' Korean TV Shows', 'Sci-Fi & Fantasy',
       ' Teen TV Shows', ' Cult Movies', 'Classic Movies', 'TV Shows',
       ' Children & Family Movies', ' Faith & Spirituality',
       ' LGBTQ Movies', 'TV Action & Adventure', 'Stand-Up Comedy',
       ' Crime TV Shows', 'Stand-Up Comedy & Talk Shows',
       'Classic & Cult TV', ' Stand-Up Comedy & Talk Shows',
       'Anime Features', ' Science & Nature TV', 'Movies',
       ' Documentaries', 'Romantic TV Shows', 'Cult Movies',
       'Independent Movies', 'TV Horror', 'Spanish-Language TV Shows',
       ' Classic & Cult TV', 'Music & Musicals', 'Romantic Movies',
       'LGBTQ Movies', ' Stand-Up Comedy', 'TV Sci-Fi & Fantasy',
       'Sports Movies'], dtype=object)
```

In [67]:
```
# I have notices there are some similiar Genre category that count two time just be
# I have removed extra sapce
df_final3["Genre"] = df_final3["Genre"].str.strip()
```

In [68]:
```
#number of distinct titles on the basis of genre
dat = df_final3.groupby("Genre").agg({"title":"nunique"}).reset_index().sort_values
```

```
plt.figure(figsize=(12,5))
sns.barplot(y=dat["Genre"],x=dat["title"],color="orange")
plt.xticks(rotation=90)
plt.xlabel('Frequency of Genres')
plt.show()
```



International Movies, Dramas, Comedies are the most popular Genre in movie

In [69]:
```
# number of distinct titles on the basis of type
```

In [70]:
```
df_final["type"].value_counts()
```

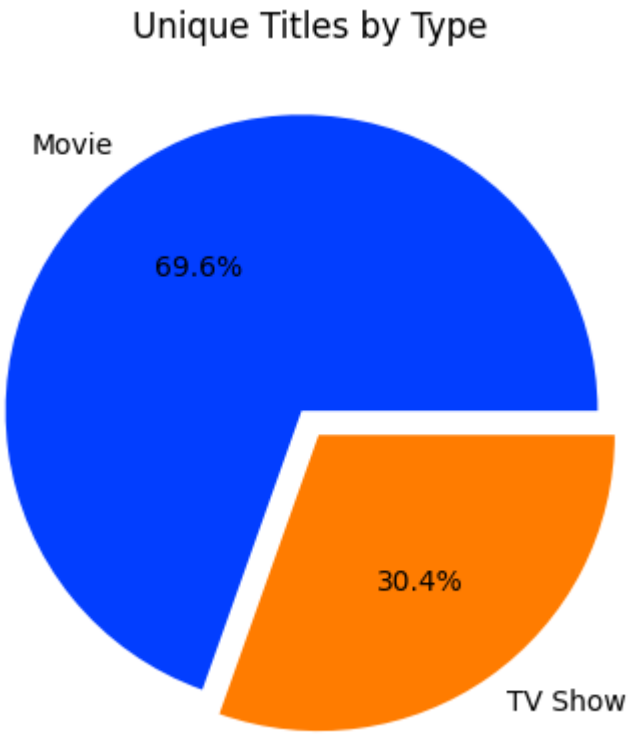
Out[70]:
```
type
Movie      145917
TV Show     56148
Name: count, dtype: int64
```

In [71]:
```
df_type = df_final3.groupby("type").agg({"title":"nunique"}).reset_index().sort_val
df_type
```

Out[71]:

	type	title
0	Movie	6115
1	TV Show	2676

In [72]:
```
 plt.pie(df_type['title'], explode=(0.05, 0.05), labels=df_type['type'], colors=sns
# Display the plot
plt.title('Unique Titles by Type')
plt.show()
```



We have 70:30 ratio of Movies and TV Shows in our data

In [73]:
```
# number of distinct titles on the basis of country
df_final3.groupby("country").agg({"title":"nunique"})
```

Out[73]:

country	title
	8
Afghanistan	1
Albania	1
Algeria	3
Angola	1
...	...
Uruguay	9
Venezuela	2
Vietnam	7
West Germany	1
Zimbabwe	1

198 rows × 1 columns

In [74]:
```
df_final3["country"] = df_final3["country"].str.replace(",","")
```

In [75]:
```
df_country = df_final3.groupby("country").agg({"title":"nunique"}).reset_index()
df_country.loc[df_country["country"].str.contains("United States"),"country"].value
```

Out[75]:
```
array([' United States', 'United States'], dtype=object)
```

The above dataframe shows a flaw in which we are seeing countries, such as Cambodia and Cambodia, or United States and United States, are shown as different countries.They should have been same

In [76]:
```
# removed Extra sapce
df_final3["country"] = df_final3["country"].str.strip()
```

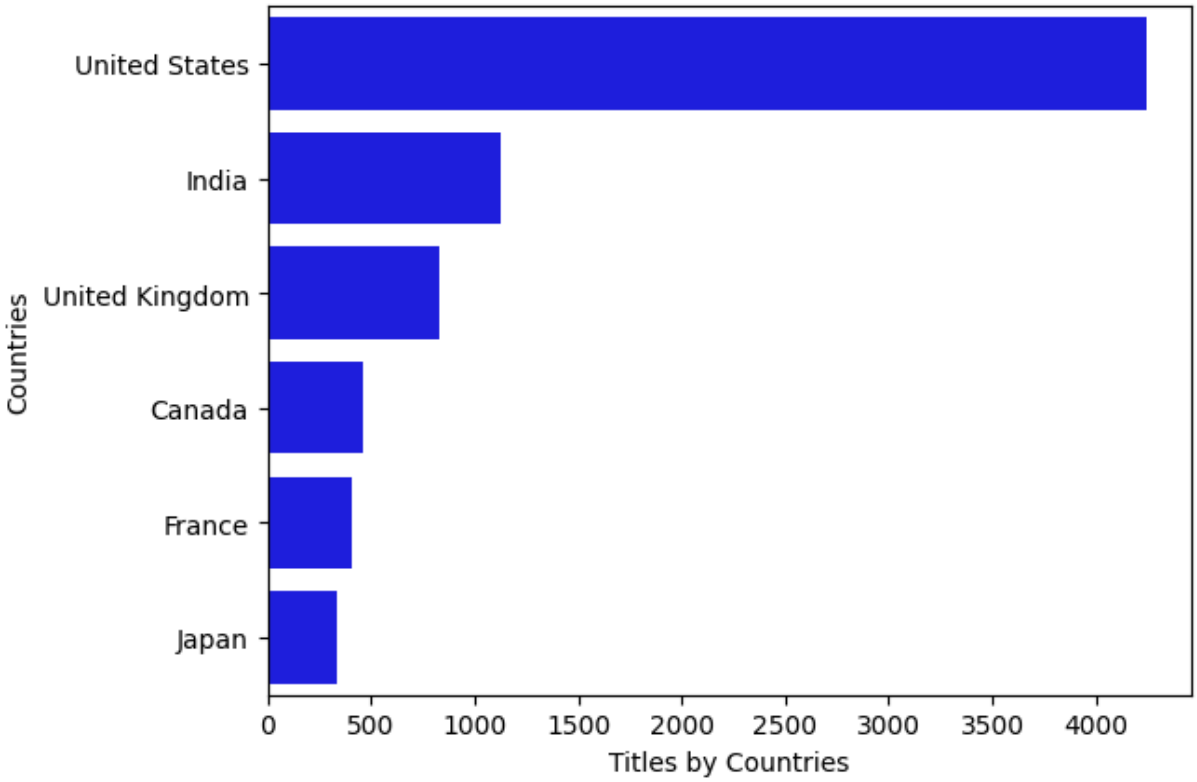
In [77]:
```
df_country = df_final3.groupby("country").agg({"title":"nunique"}).reset_index().so
df_country.head(3)
```

Out[77]:

	country	title
116	United States	4246
46	India	1123
115	United Kingdom	829

In [78]:
```
sns.barplot(x=df_country[:6]["title"],y=df_country[:6]["country"],color="blue")
plt.xlabel('Titles by Countries')
```

```
plt.ylabel('Countries')
plt.show()
```



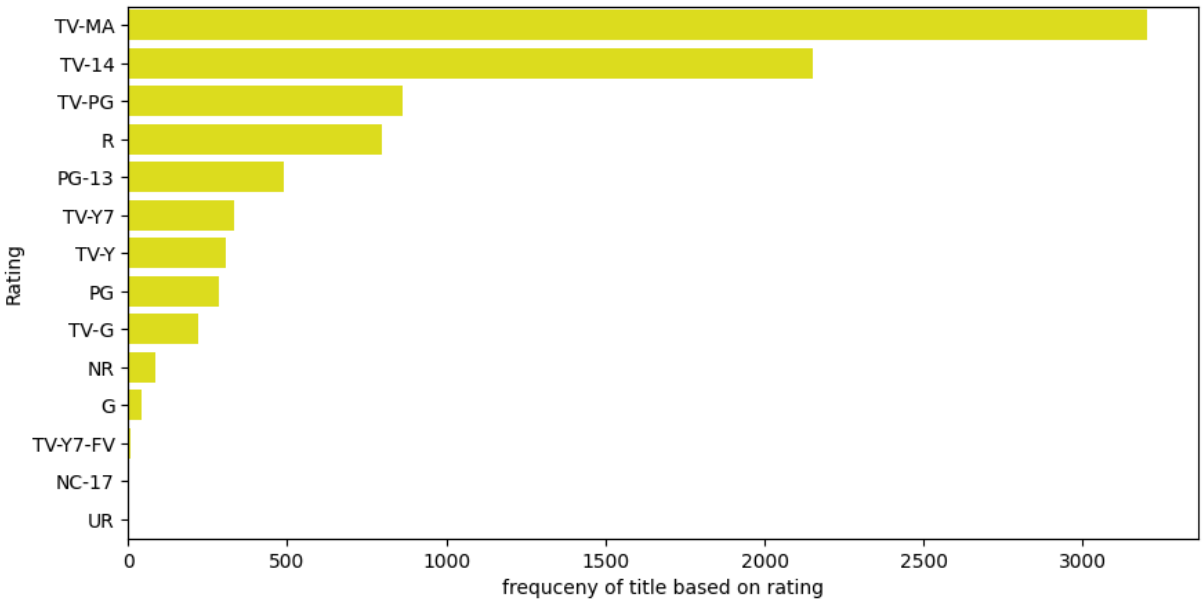
US,India,UK,Canada and France are leading countries in Content Creation on Netflix

In [79]:
```
#number of distinct titles on the basis of rating
df_rating = df_final3.groupby("rating").agg({"title":"nunique"}).reset_index().sort
df_rating
```

Out[79]:

	rating	title
8	TV-MA	3204
6	TV-14	2151
9	TV-PG	863
5	R	799
4	PG-13	490
11	TV-Y7	334
10	TV-Y	305
3	PG	287
7	TV-G	220
2	NR	87
0	G	41
12	TV-Y7-FV	6
1	NC-17	3
13	UR	3

In [80]:
```
plt.figure(figsize=(10,5))
sns.barplot(x=df_rating["title"],y=df_rating["rating"],color="yellow")
plt.xlabel("frequceny of title based on rating")
plt.ylabel("Rating")
plt.show()
```



Most of the highly rated content on Netflix is intended for Mature Audiences, R Rated, content not intended for audience under 14 and those which require Parental Guidance

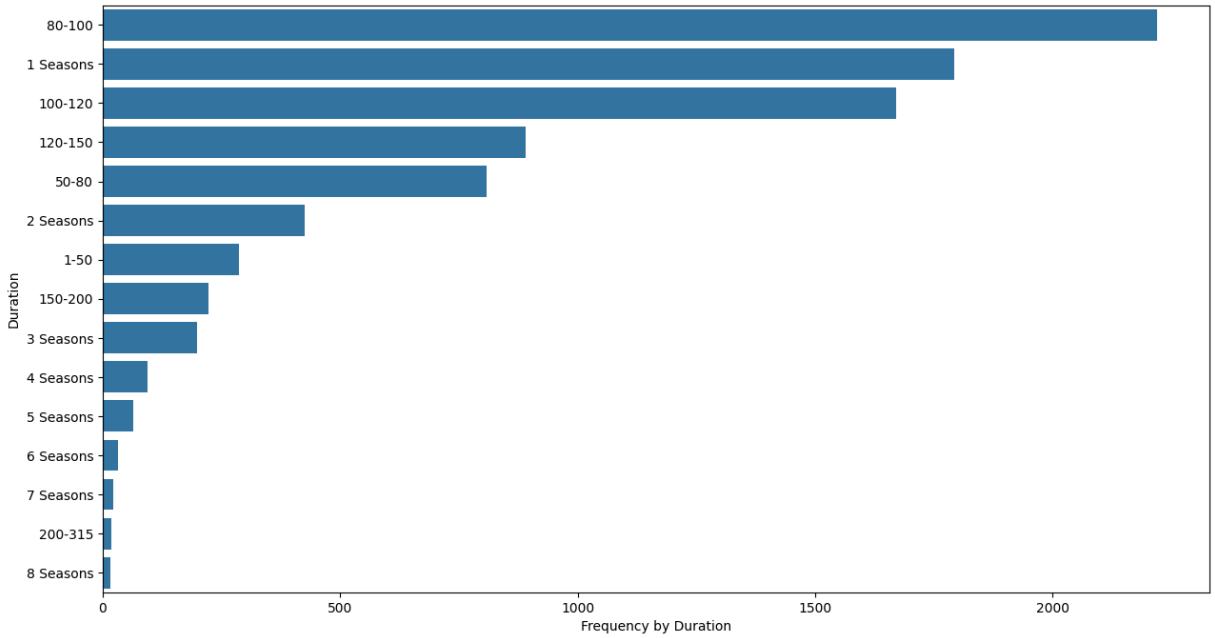
In [81]:
```
df_final3.loc[df_final3["duration"]=="<1","duration"] = df_final3.loc[df_final3["du
```

In [82]:
```
#number of distinct titles on the basis of duration
df_duration = df_final3.groupby(['duration']).agg({"title":"nunique"}).reset_index(
df_duration
```

Out[82]:

	duration	title
20	80-100	2220
0	1 Seasons	1793
3	100-120	1671
6	120-150	891
16	50-80	808
11	2 Seasons	425
1	1-50	287
9	150-200	222
13	3 Seasons	199
14	4 Seasons	95
15	5 Seasons	65
17	6 Seasons	33
18	7 Seasons	23
12	200-315	19
19	8 Seasons	17

In [83]:
```
plt.figure(figsize=(15,8))
sns.barplot(y=df_duration['duration'], x=df_duration['title'])
plt.xlabel('Frequency by Duration')
plt.ylabel('Duration')
plt.show()
```



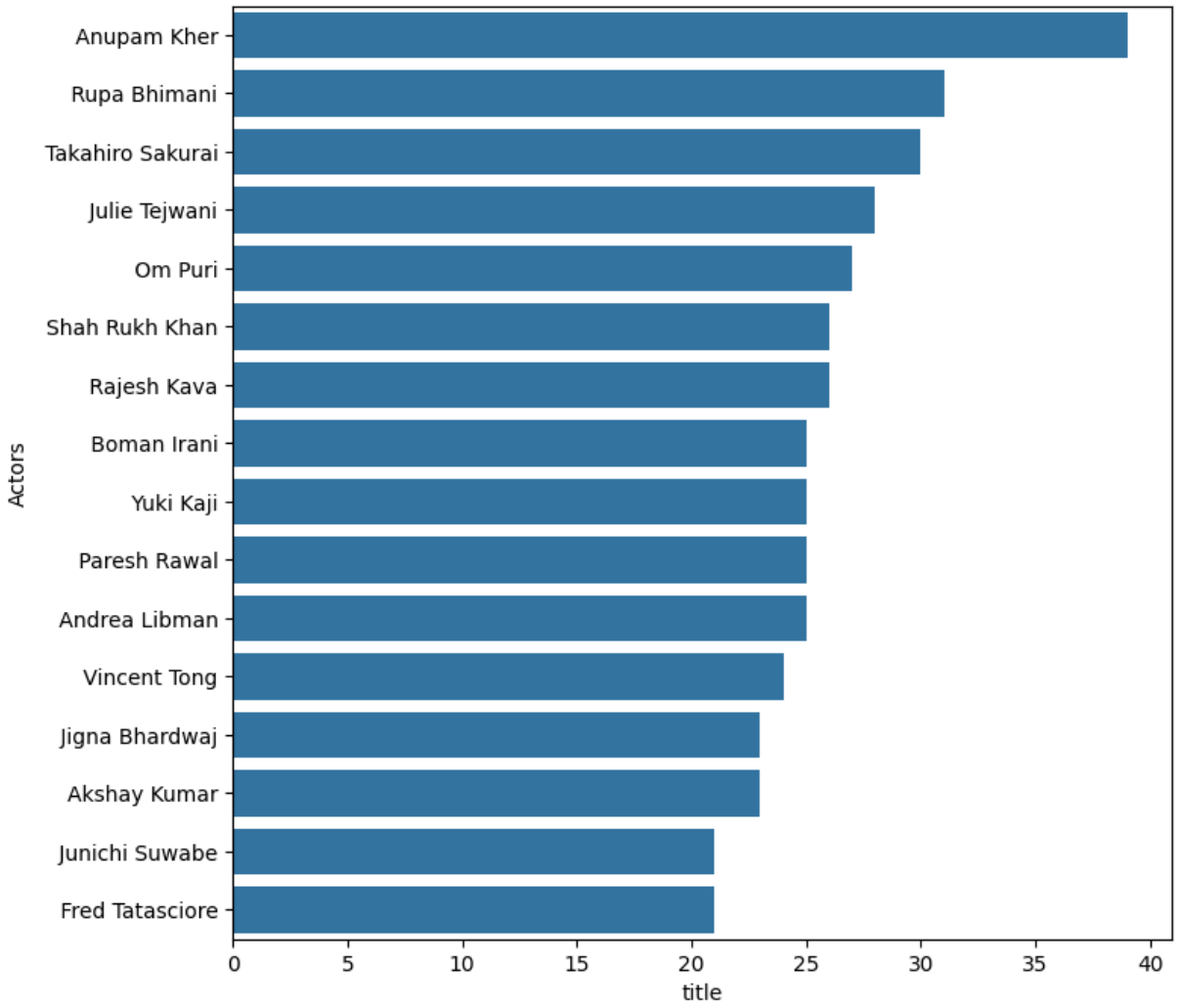
The duration of Most Watched content in our whole data is 80-100 mins.These must be movies and Shows having only 1 Season.

In [84]:
```
#number of distinct titles on the basis of Actors
df_actor = df_final3.loc[df_final3["Actors"]!="Unknow Actors"].groupby("Actors").ag
df_actor.head(3)
```

Out[84]:

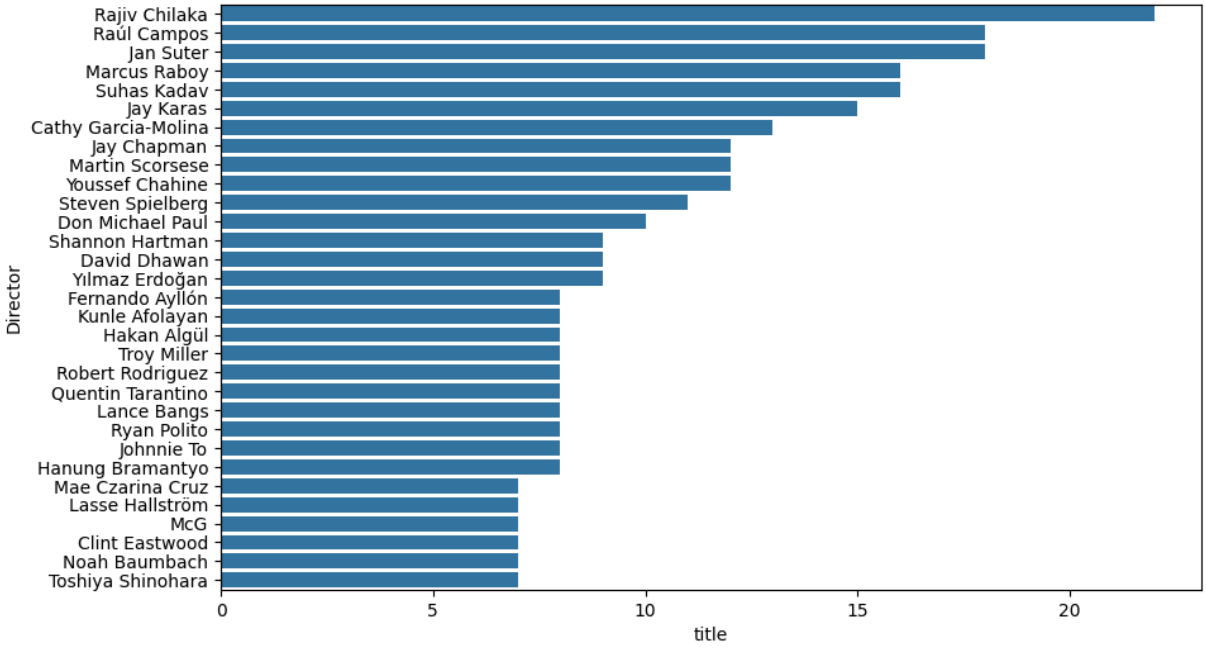
	Actors	title
2612	Anupam Kher	39
26941	Rupa Bhimani	31
30303	Takahiro Sakurai	30

In [85]:
```
plt.figure(figsize=(8,8))
sns.barplot(x=df_actor["title"],y=df_actor["Actors"])
plt.show()
```



Anupam Kher,SRK,Julie Tejwani, Naseeruddin Shah and Takahiro Sakurai occupy the top stop in Most Watched content.

In [86]:
```
#number of distinct titles on the basis of Actors
df_directors = df_final3.loc[df_final3["Director"]!="Unknow Director"].groupby(['Di
plt.figure(figsize=(10,6))
sns.barplot(x=df_directors["title"],y=df_directors["Director"])
plt.show()
```



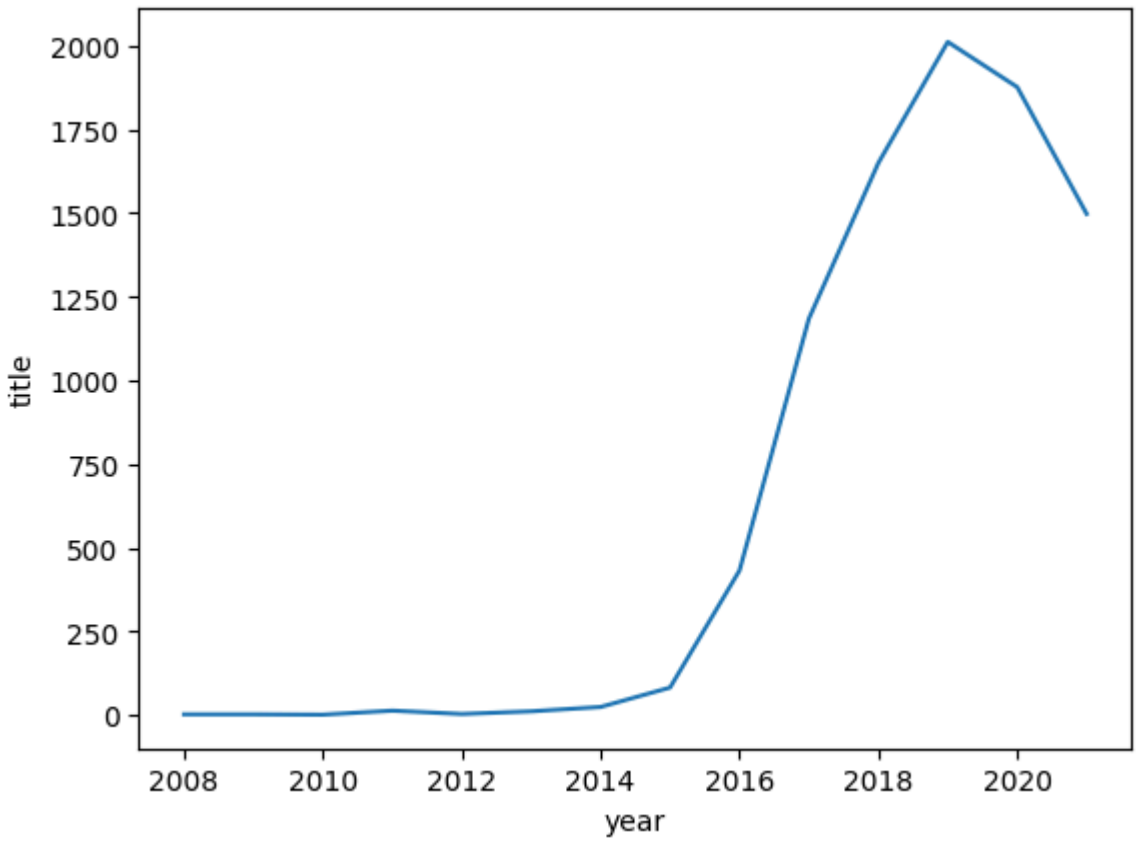
Rajiv Chilaka, Jan Suter and Raul Campos are the most popular directors across Netflix

In [87]:
```
df_year = df_final3.groupby("year").agg({"title":"nunique"}).reset_index().sort_val
df_year
```

Out[87]:

	year	title
11	2019	2012
12	2020	1877
10	2018	1650
13	2021	1498
9	2017	1185
8	2016	432
7	2015	82
6	2014	24
3	2011	13
5	2013	11
4	2012	3
0	2008	2
1	2009	2
2	2010	1

In [88]:
```
sns.lineplot(y=df_year["title"],x=df_year["year"])
plt.show()
```



The Amount of Content across Netflix has increased from 2008 continuously till 2019. Then started decreasing from here(probably due to Covid)

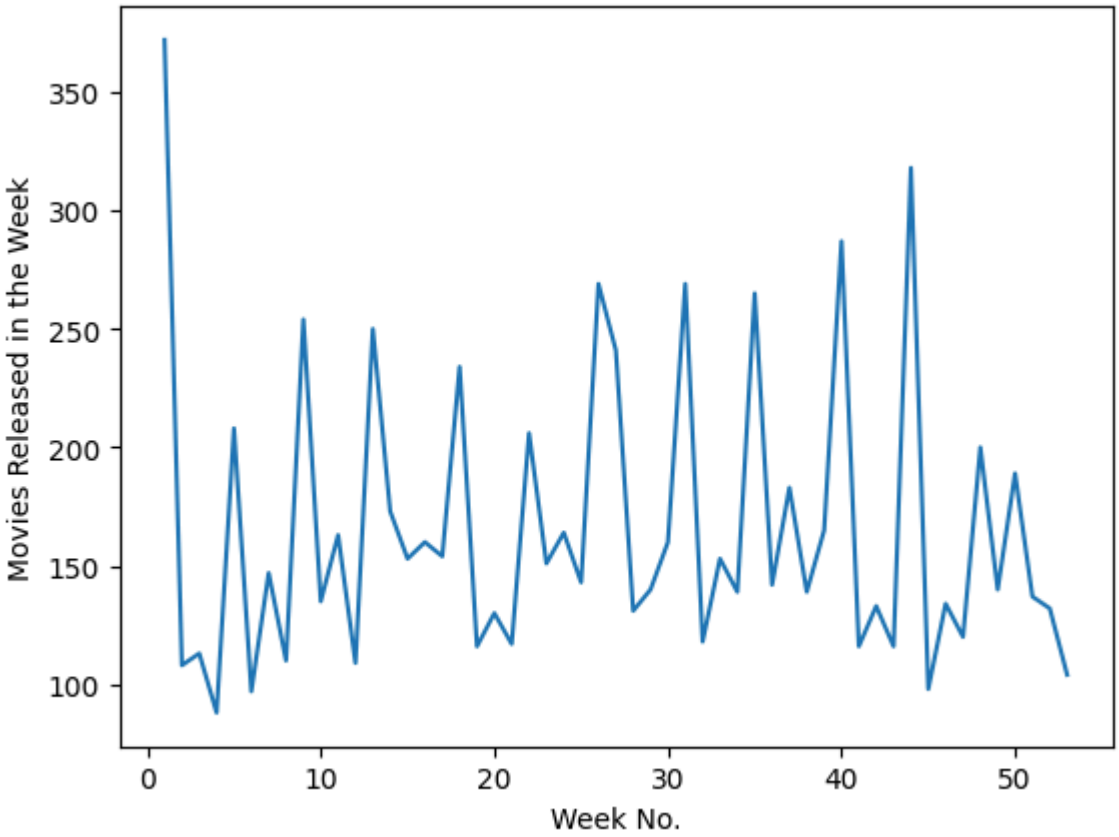
In [89]:
```
#number of distinct titles on the basis of week
df_week = df_final3.groupby("week").agg({"title":"nunique"}).reset_index().sort_val
df_week
```

Out[89]:

	week	title
0	1	372
43	44	318
39	40	287
30	31	269
25	26	269
34	35	265
8	9	254
12	13	250
26	27	241
17	18	234
4	5	208
21	22	206
47	48	200
49	50	189
36	37	183
13	14	173
38	39	165
23	24	164
10	11	163
15	16	160
29	30	160
16	17	154
32	33	153
14	15	153
22	23	151
6	7	147
24	25	143
35	36	142
48	49	140
28	29	140

	week	title
33	34	139
37	38	139
50	51	137
9	10	135
45	46	134
41	42	133
51	52	132
27	28	131
19	20	130
46	47	120
31	32	118
20	21	117
40	41	116
42	43	116
18	19	116
2	3	113
7	8	110
11	12	109
1	2	108
52	53	104
44	45	98
5	6	97
3	4	88

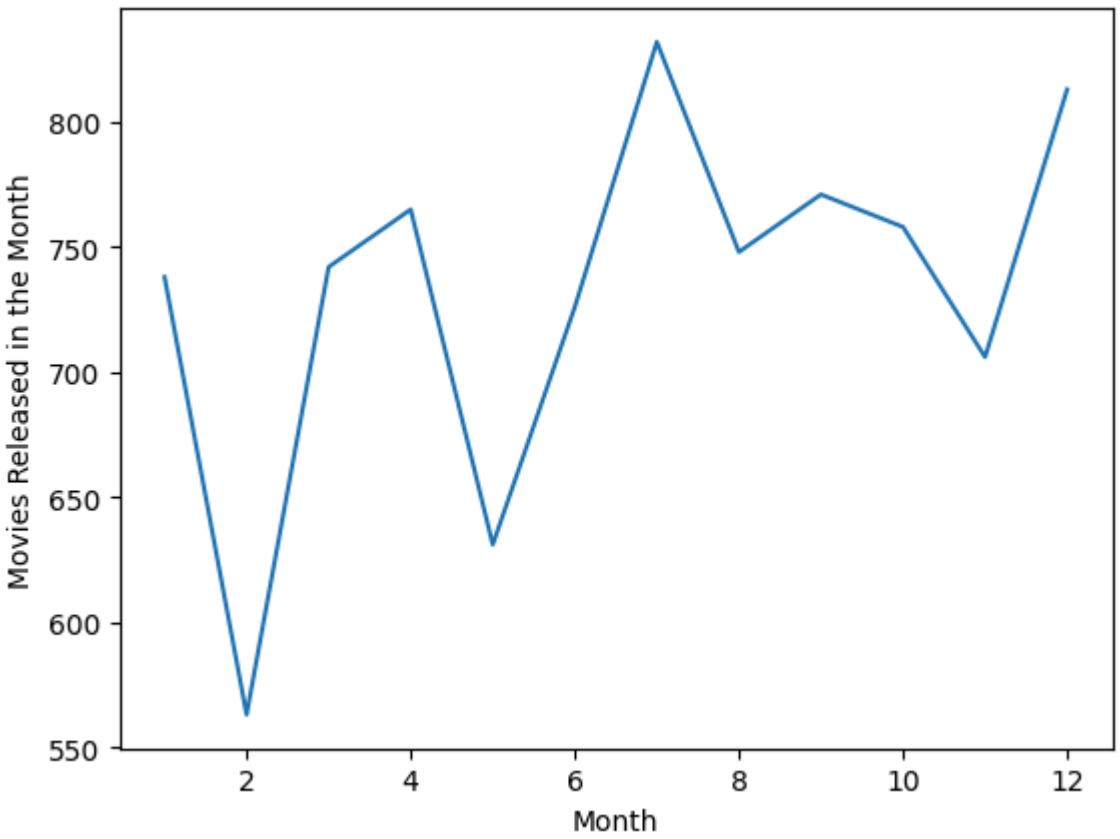
In [90]:
```
sns.lineplot(df_week,x="week",y="title")
plt.ylabel("Movies Released in the Week")
plt.xlabel("Week No.")
plt.show()
```



Most of the Content across Netflix is added in the first week of the year and it follows a bit of a cyclical pattern

In [91]:
```
#number of distinct titles on the basis of week
df_month = df_final3.groupby(['month']).agg({"title":"nunique"}).reset_index().sort
```

In [92]:
```
sns.lineplot(data=df_month, x='month', y='title')
plt.ylabel("Movies Released in the Month")
plt.xlabel("Month")
plt.show()
```



Most of the content is added in the first and last months across Netflix(reinstating what we observed for first week in baove plot)

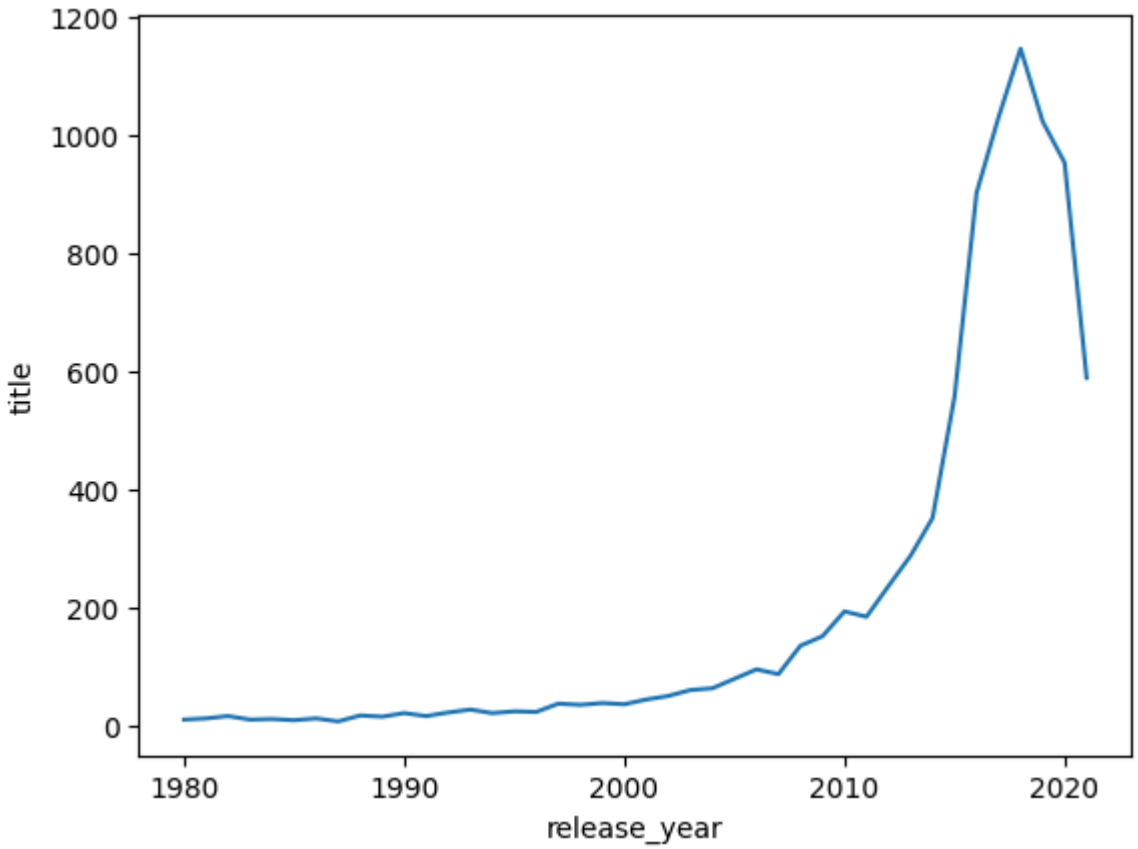
In [93]:
```
df_movie_year = df_final3.loc[df_final3["release_year"]>=1980].groupby("release_yea
df_movie_year
```

Out[93]:

	release_year	title
38	2018	1146
37	2017	1030
39	2019	1023
40	2020	953
36	2016	902
41	2021	589
35	2015	557
34	2014	352
33	2013	288
32	2012	237
30	2010	194
31	2011	185
29	2009	152
28	2008	136
26	2006	96
27	2007	88
25	2005	80
24	2004	64
23	2003	61
22	2002	51
21	2001	45
19	1999	39
17	1997	38
20	2000	37
18	1998	36
13	1993	28
15	1995	25
16	1996	24
12	1992	23
14	1994	22

	release_year	title
10	1990	22
8	1988	18
11	1991	17
2	1982	17
9	1989	16
6	1986	13
1	1981	13
4	1984	12
3	1983	11
0	1980	11
5	1985	10
7	1987	8

In [94]:
```
sns.lineplot(data=df_movie_year,x="release_year",y="title")
plt.show()
```



Net content release which are later uploaded to Netflix has increased since 1980 till 2020 though later reduced certainly due to COVID-19

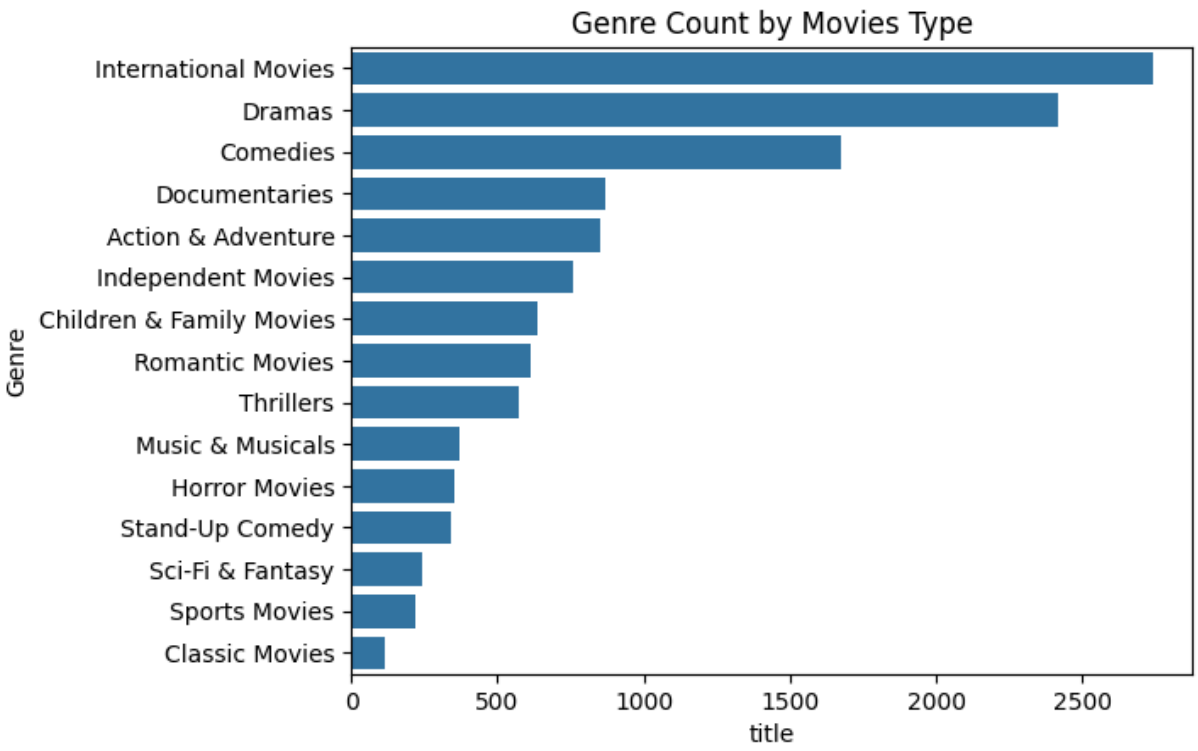
Univariate Analysis separately for shows and movies

In [95]:
```
df_final["type"].value_counts()
```

Out[95]:
```
type
Movie      145917
TV Show     56148
Name: count, dtype: int64
```

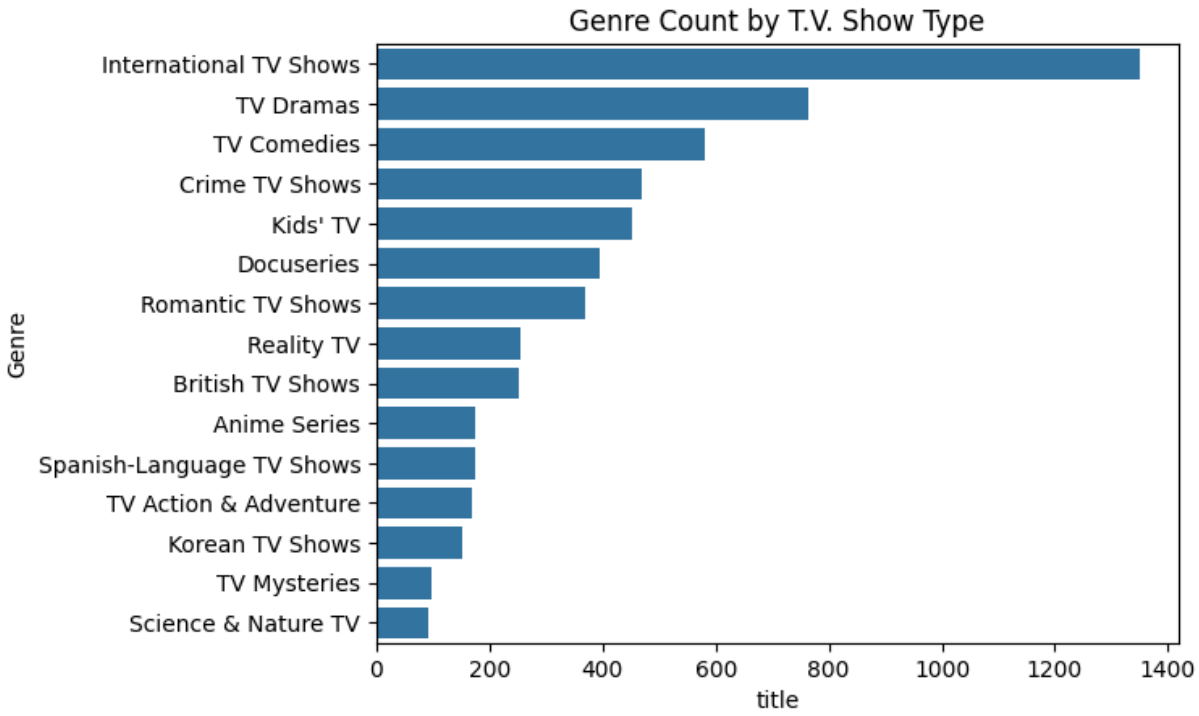
In [96]:
```
df_movie = df_final3.loc[df_final3["type"]=="Movie"]
df_show  = df_final3.loc[df_final3["type"]=="TV Show"]
```

In [97]:
```
df_ = df_movie.groupby("Genre").agg({"title":"nunique"}).reset_index().sort_values(
sns.barplot(df_,x="title",y="Genre")
plt.title("Genre Count by Movies Type")
plt.show()
```



International Movies, Dramas and Comedy Genres are popular across Movies in Netflix

In [98]:
```
df__ = df_show.groupby("Genre").agg({"title":"nunique"}).reset_index().sort_values(
sns.barplot(df__,x="title",y="Genre")
plt.title("Genre Count by T.V. Show Type")
plt.show()
```



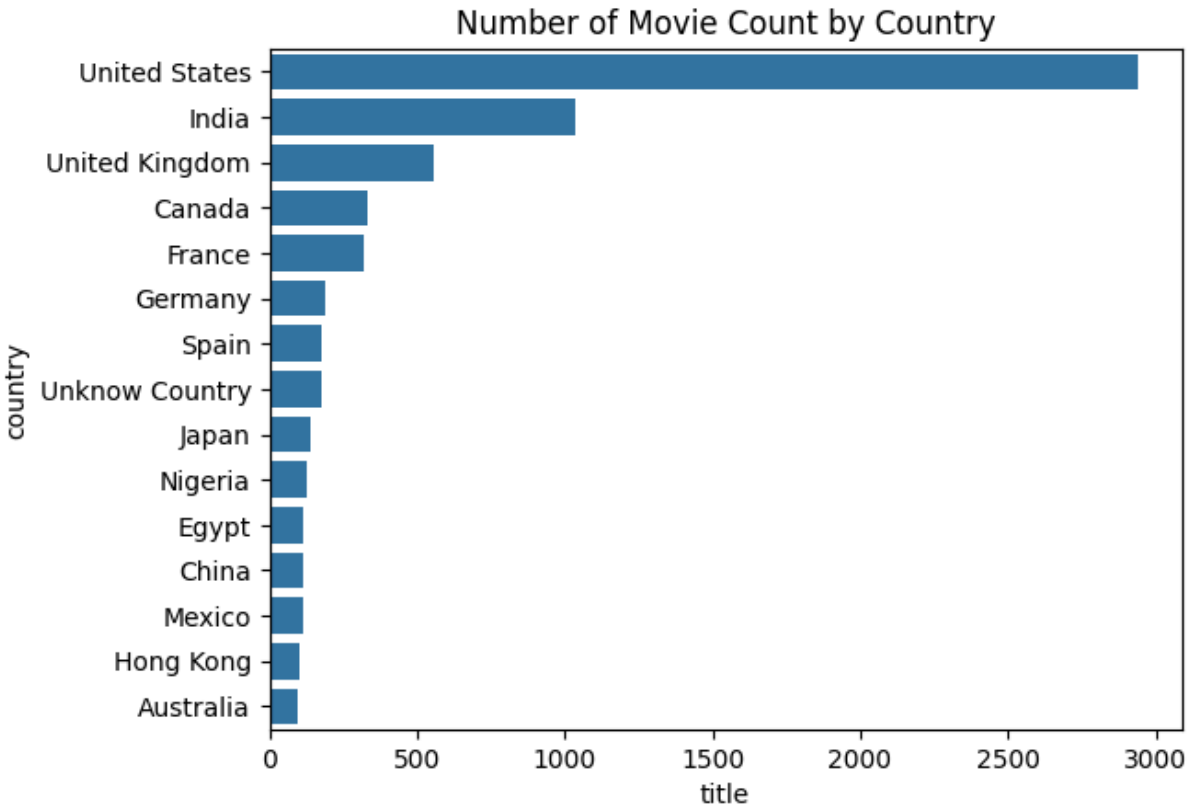
International Show, Dramas and Comedy Genres are popular followed by Documentaries across Show on Netflix

In [99]:
```
dfm_country = df_movie.groupby("country").agg({"title":"nunique"}).reset_index().so
dfm_country.head(3)
```

Out[99]:

	country	title
111	United States	2940
42	India	1037
110	United Kingdom	556

In [100…
```
sns.barplot(y="country",x="title",data=dfm_country[:15])
plt.title("Number of Movie Count by Country")
plt.show()
```

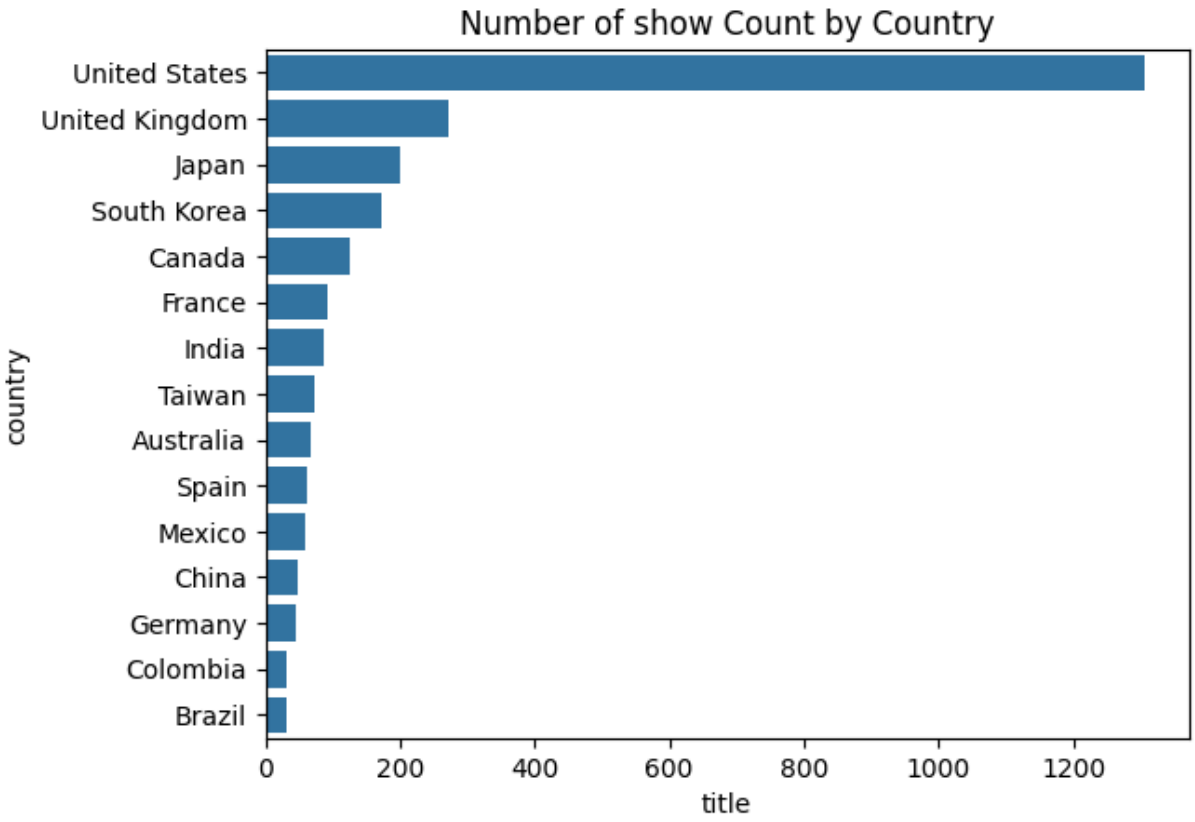


In [101…
```
dfs_country = df_show.groupby("country").agg({"title":"nunique"}).reset_index().sor
dfs_country.head(3)
```

Out[101…

	country	title
64	United States	1306
63	United Kingdom	273
31	Japan	200

In [102…
```
sns.barplot(y="country",x="title",data=dfs_country[:15])
plt.title("Number of show Count by Country")
plt.show()
```



United States is leading across both TV Shows and Movies, UK also provides great content across TV Shows and Movies. Surprisingly India is much more prevalent in Movies as compared TV Shows.

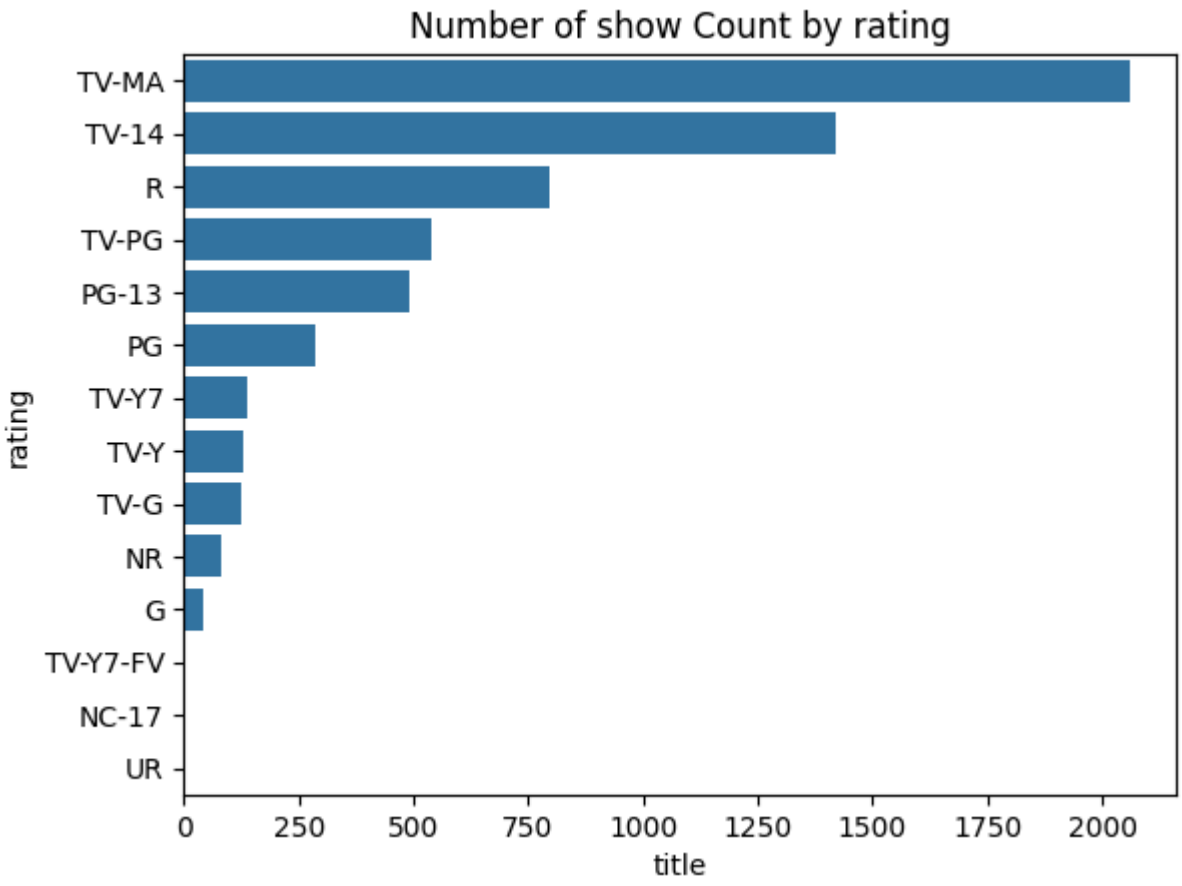
Moreover the number of Movies created in India outweigh the sum of TV Shows and Movies across UK since India was rated as second in net sum of whole content across Netflix.

In [103…
```
dfm_rating = df_movie.groupby("rating").agg({"title":"nunique"}).reset_index().sort
dfm_rating.head(3)
```

Out[103…

	rating	title
8	TV-MA	2059
6	TV-14	1418
5	R	797

In [104…
```
sns.barplot(y="rating",x="title",data=dfm_rating[:15])
plt.title("Number of show Count by rating")
plt.show()
```

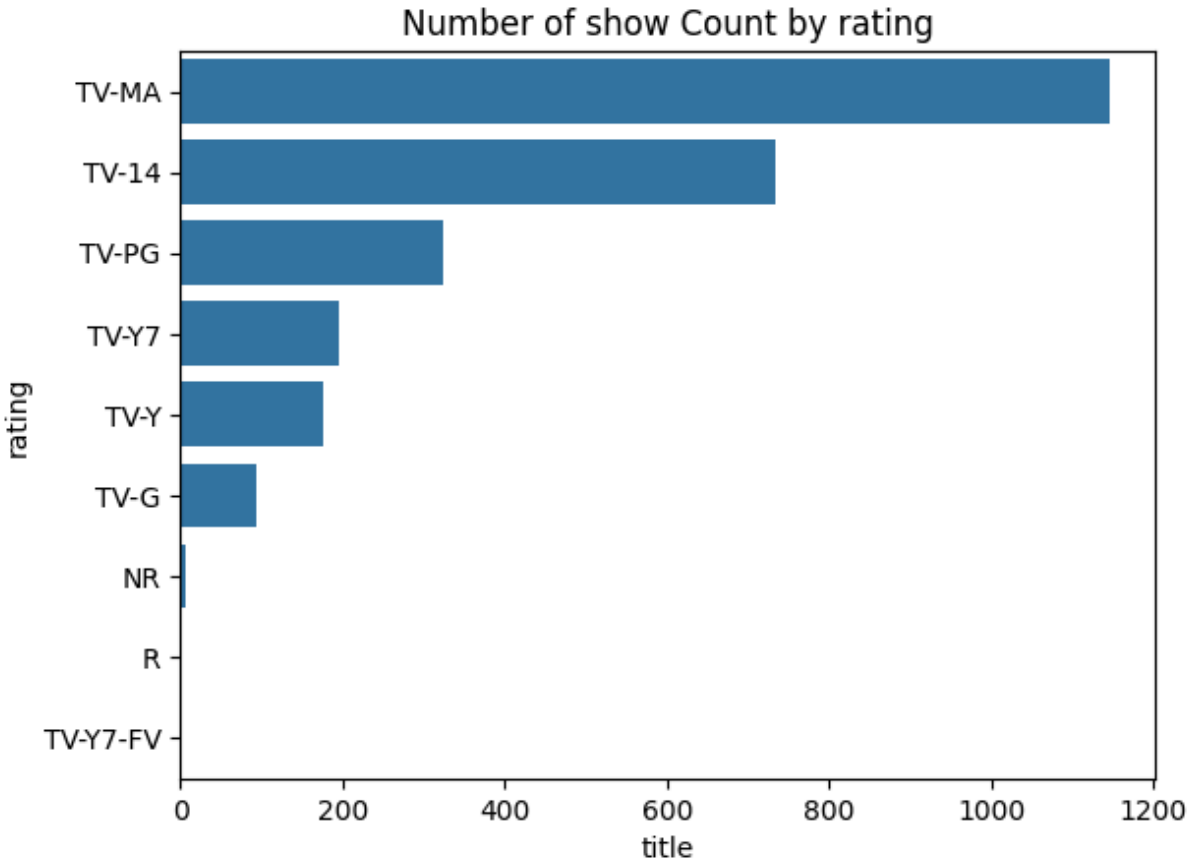


In [105…
```
dfs_rating = df_show.groupby("rating").agg({"title":"nunique"}).reset_index().sort_
dfs_rating.head(3)
```

Out[105…

	rating	title
4	TV-MA	1145
2	TV-14	733
5	TV-PG	323

In [106…
```
sns.barplot(y="rating",x="title",data=dfs_rating[:15])
plt.title("Number of show Count by rating")
plt.show()
```



So it seems plaussible to conclude that the popular ratings across Netflix includes Mature Audiences and those appropriate for over 14/over 17 ages.

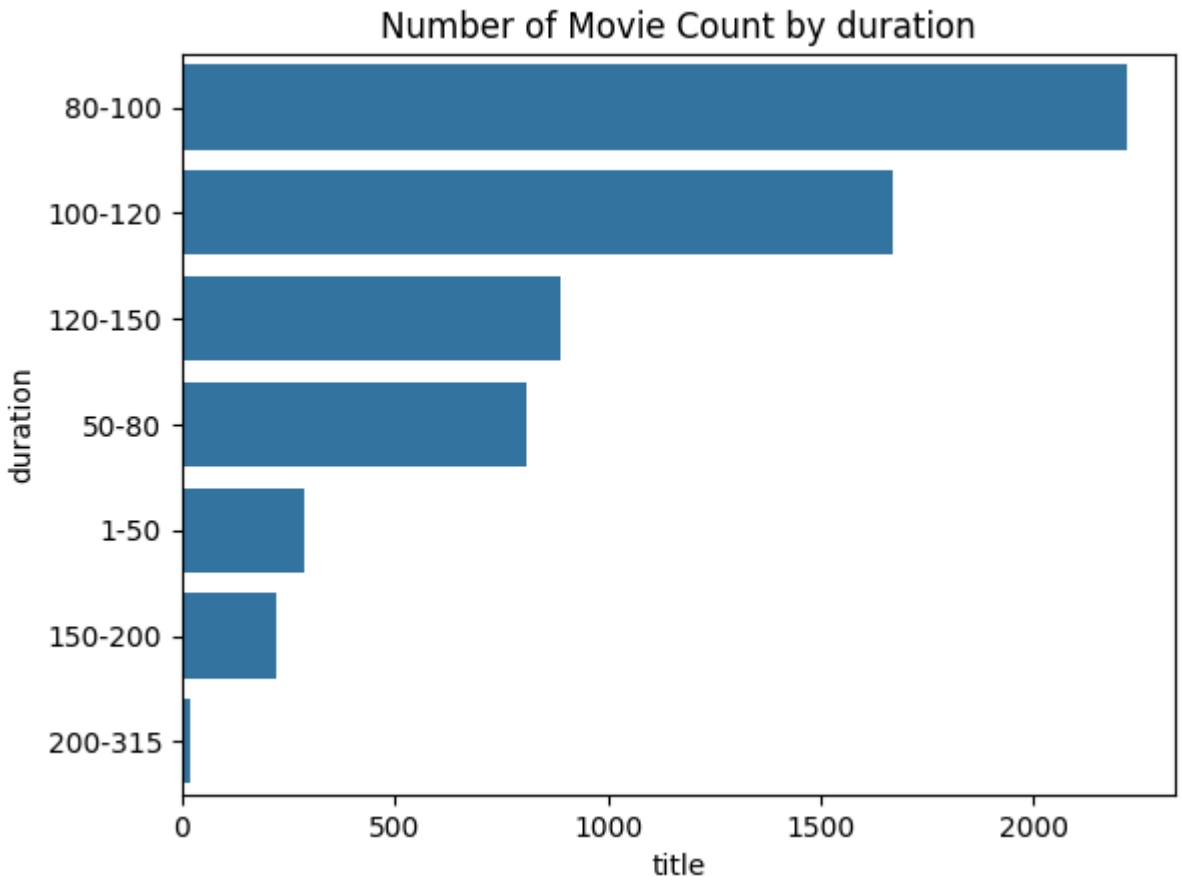
Moreover there are no TV Shows having a rating of R

In [107…
```
dfm_duration = df_movie.groupby("duration").agg({"title":"nunique"}).reset_index().
dfm_duration.head(3)
```

Out[107…

	duration	title
6	80-100	2220
1	100-120	1671
2	120-150	891

In [108…
```
sns.barplot(y="duration",x="title",data=dfm_duration[:15])
plt.title("Number of Movie Count by duration")
plt.show()
```



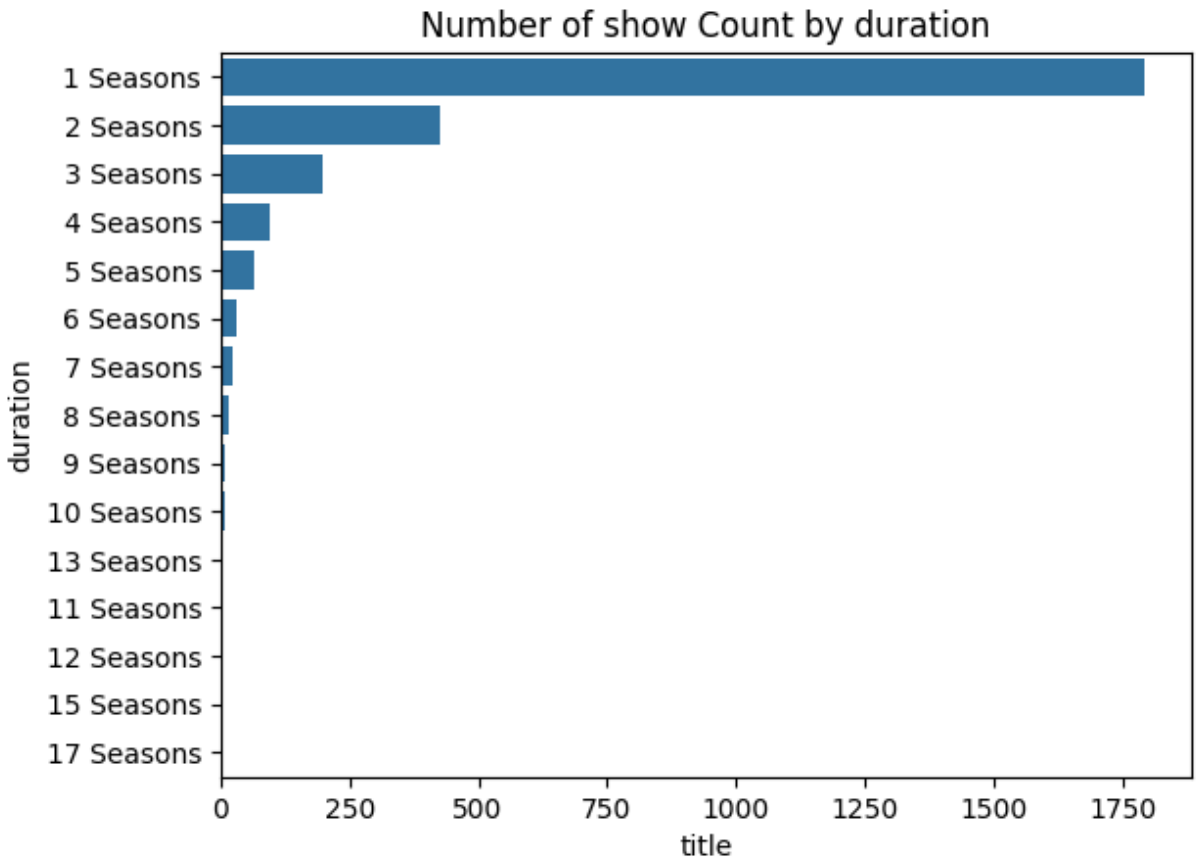
Across movies 80-100,100-120 and 120-150 is the ranges of minutes for which most movies lie. So quite possibly 80-150 mins is the sweet spot we would be wanting for movies.

In [109…
```
dfs_duration = df_show.groupby("duration").agg({"title":"nunique"}).reset_index().s
dfs_duration.head(3)
```

Out[109…

	duration	title
0	1 Seasons	1793
7	2 Seasons	425
8	3 Seasons	199

In [110…
```
sns.barplot(y="duration",x="title",data=dfs_duration[:15])
plt.title("Number of show Count by duration")
plt.show()
```



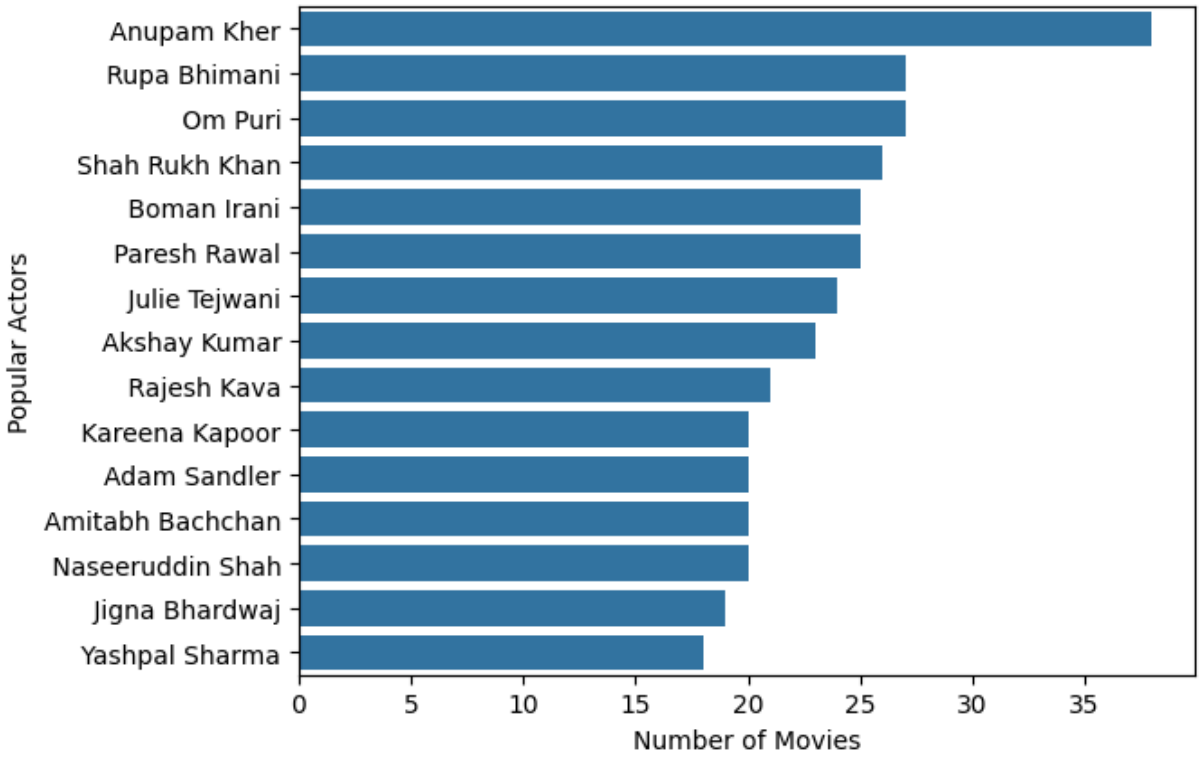
Across TV Shows, shows having only 1 Season are common as soon as the season length increases, the number of shows decrease and this definitely sounds as expected

In [114…
```
dfm_actor = df_movie.loc[df_movie["Actors"]!="Unknow Actors"].groupby("Actors").agg
dfm_actor.head(3)
```

Out[114…

	Actors	title
1946	Anupam Kher	38
19235	Rupa Bhimani	27
16781	Om Puri	27

In [123…
```
sns.barplot(y="Actors",x="title",data=dfm_actor)
plt.xlabel('Number of Movies')
plt.ylabel('Popular Actors')
plt.show()
```



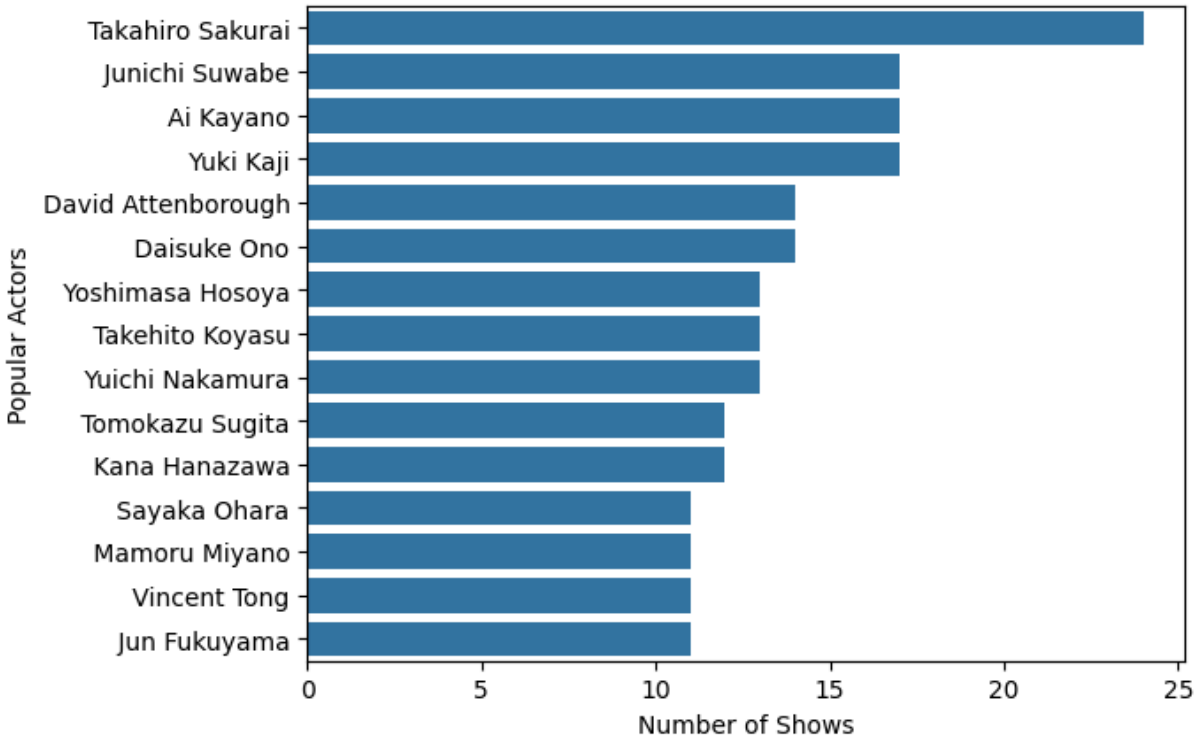
Our bollywood actors such as Anupam Kher, SRK, Naseeruddin Shah are very much popular acrossmovies on Netflix

In [124...
```
dfs_actor = df_show.loc[df_show["Actors"]!="Unknow Actors"].groupby("Actors").agg({
dfs_actor.head(3)
```

Out[124...

	Actors	title
11974	Takahiro Sakurai	24
6136	Junichi Suwabe	17
222	Ai Kayano	17

In [125...
```
sns.barplot(y="Actors",x="title",data=dfs_actor)
plt.xlabel('Number of Shows')
plt.ylabel('Popular Actors')
plt.show()
```



Takahiro Sakurai,Yuki Kaji and other South Korean/Japanese actors are the most popular actors across TV Shows

In [126…

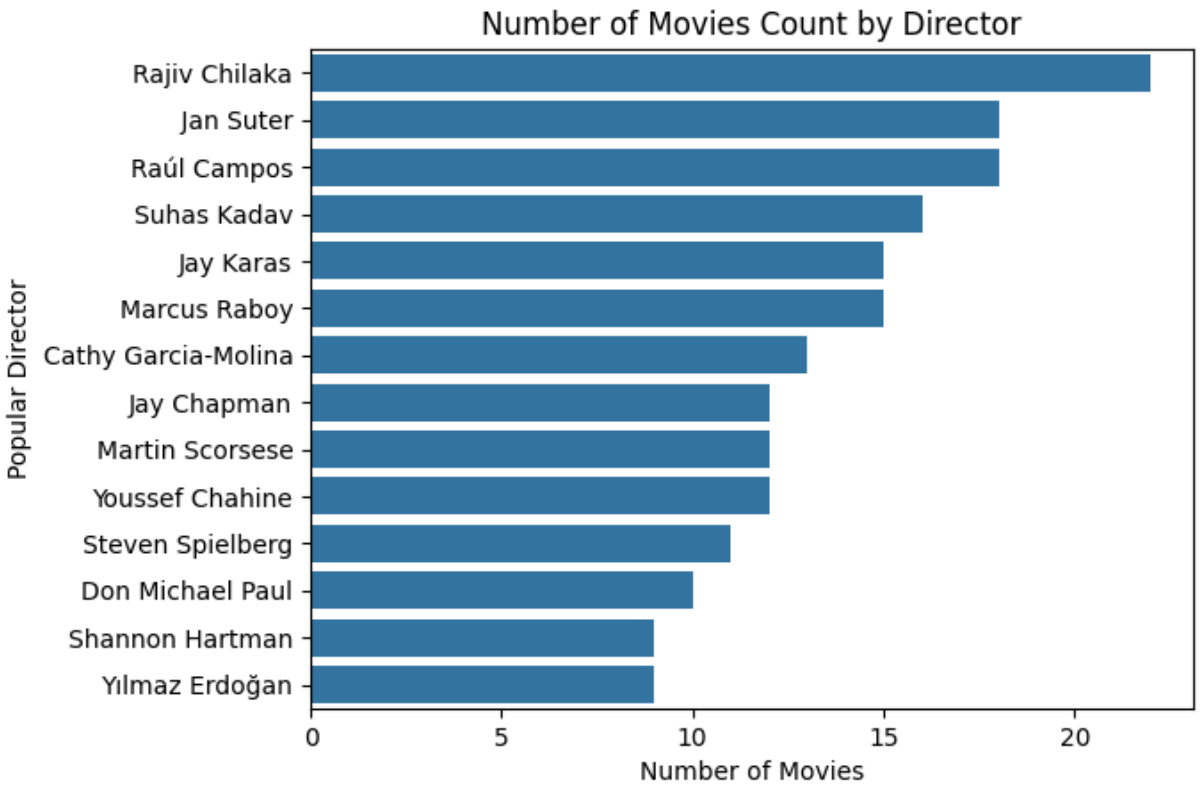
```
dfm_director = df_movie.groupby("Director").agg({"title":"nunique"}).reset_index().
dfm_director.head(3)
```

Out[126…

	Director	title
3818	Rajiv Chilaka	22
234	Jan Suter	18
3865	Raúl Campos	18

In [127…

```
sns.barplot(y="Director",x="title",data=dfm_director)
plt.title("Number of Movies Count by Director")
plt.xlabel('Number of Movies')
plt.ylabel('Popular Director')
plt.show()
```



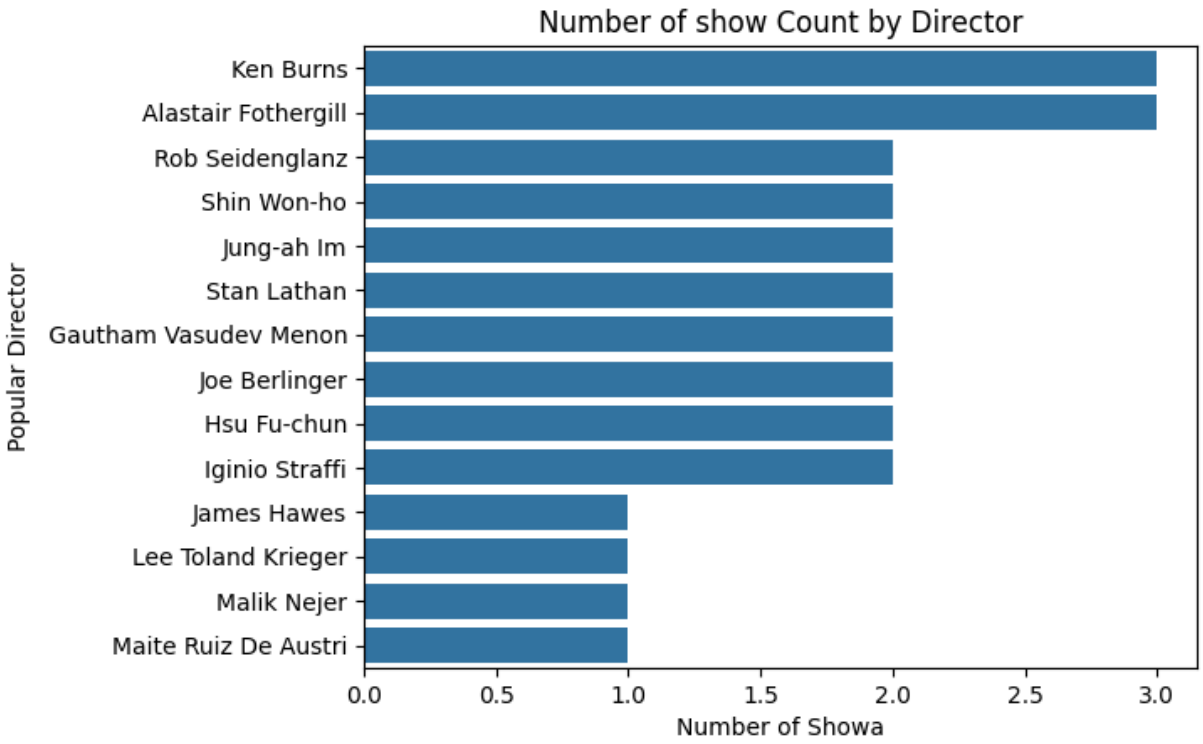
Rajiv Chilka, Jan Suter, Raul Campos, Suhas Kadav are popular directors across movies

In [129…
```
dfs_director = df_show.groupby("Director").agg({"title":"nunique"}).reset_index().s
dfs_director.head(3)
```

Out[129…

	Director	title
187	Ken Burns	3
86	Alastair Fothergill	3
255	Rob Seidenglanz	2

In [130…
```
sns.barplot(y="Director",x="title",data=dfs_director)
plt.title("Number of show Count by Director")
plt.xlabel('Number of Showa')
plt.ylabel('Popular Director')
plt.show()
```



Ken Burns, Alastair Fothergill, Stan Lathan, Joe Barlinger are popular directors across TV Shows on Netflix

In [131…

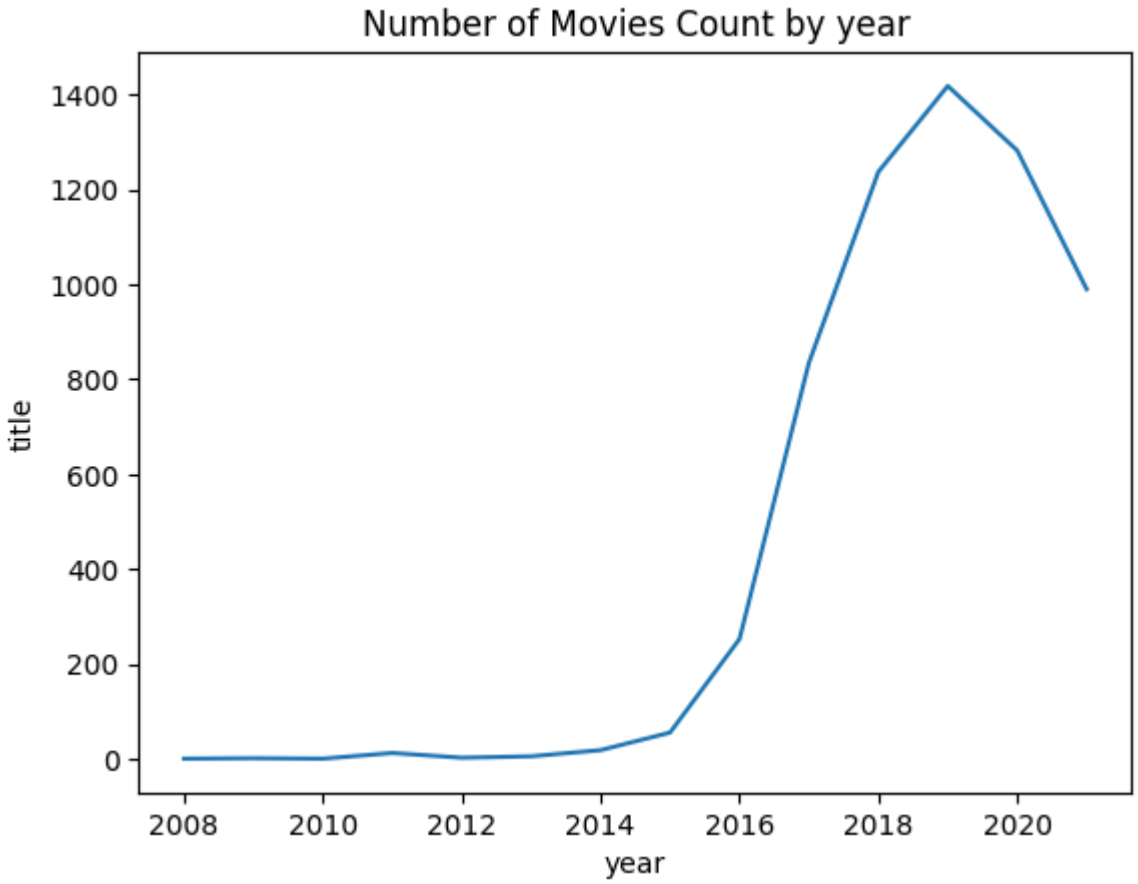
```
dfm_year = df_movie.groupby("year").agg({"title":"nunique"}).reset_index().sort_val
dfm_year.head(3)
```

Out[131…

	year	title
11	2019	1418
12	2020	1282
10	2018	1237

In [132…

```
sns.lineplot(x="year",y="title",data=dfm_year)
plt.title("Number of Movies Count by year")
plt.show()
```

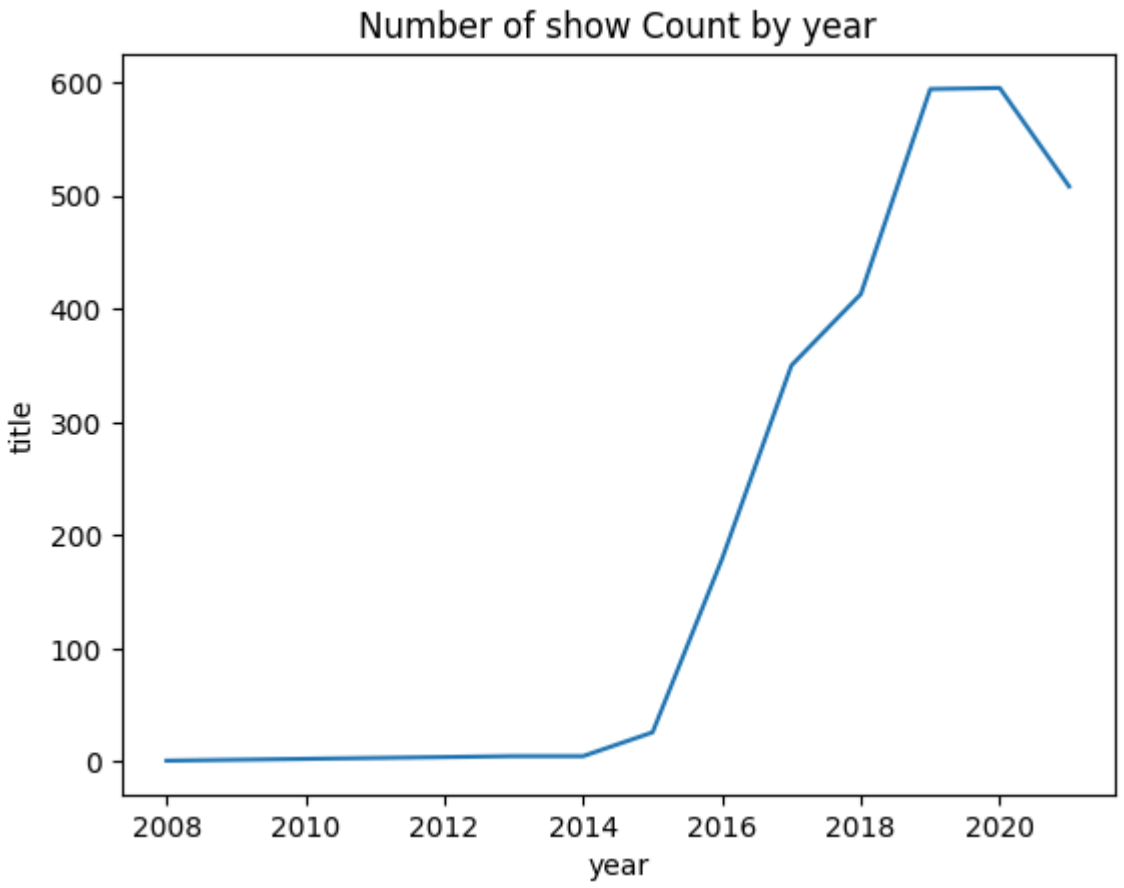


In [133…
```
dfs_year = df_show.groupby("year").agg({"title":"nunique"}).reset_index().sort_valu
dfs_year.head(3)
```

Out[133…

	year	title
8	2020	595
7	2019	594
9	2021	508

In [134…
```
sns.lineplot(x="year",y="title",data=dfs_year)
plt.title("Number of show Count by year")
plt.show()
```



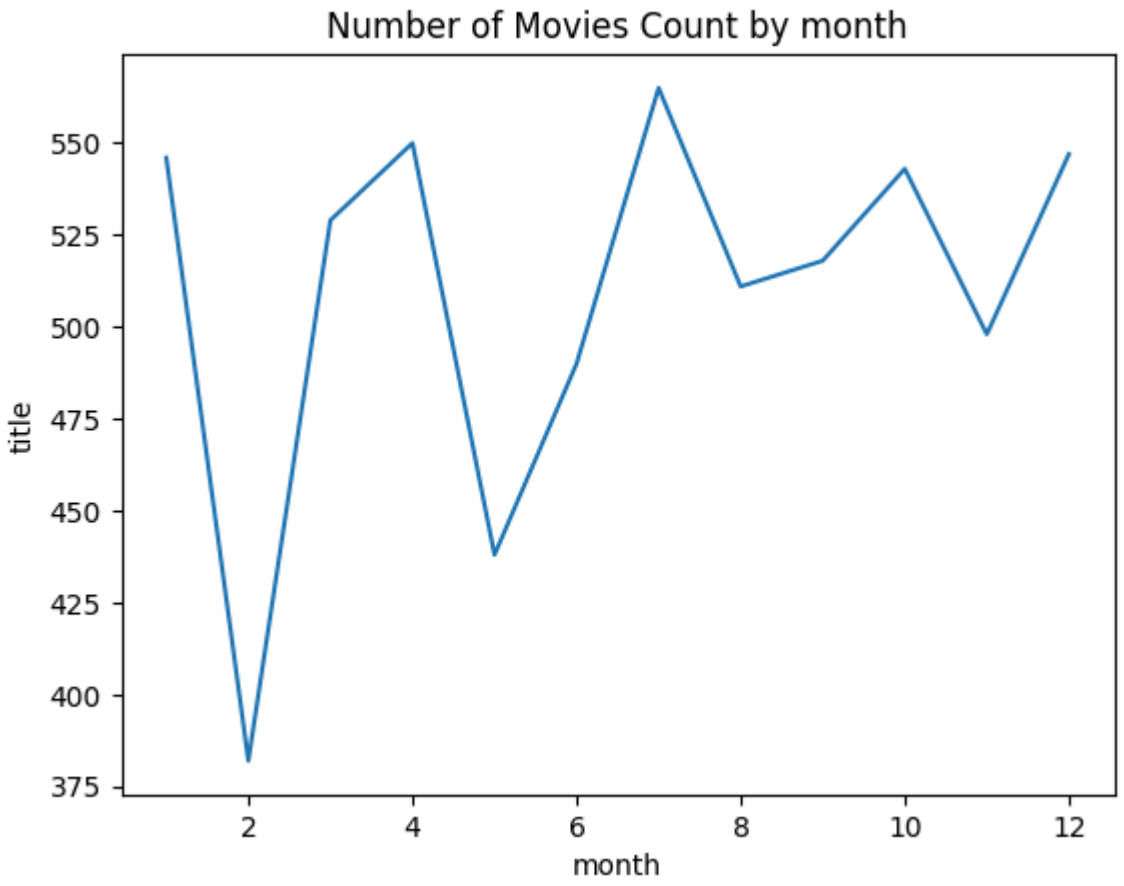
Till 2019, overall content across Netflix was increasing but due to Covid in 2020, though TV Shows didn't take a hit then Movies did take a hit. Well later in 2021, content across both was reduced significantly

In [135…
```
dfm_month = df_movie.groupby("month").agg({"title":"nunique"}).reset_index().sort_v
dfm_month.head(3)
```

Out[135…

	month	title
6	7	565
3	4	550
11	12	547

In [136…
```
sns.lineplot(x="month",y="title",data=dfm_month)
plt.title("Number of Movies Count by month")
plt.show()
```

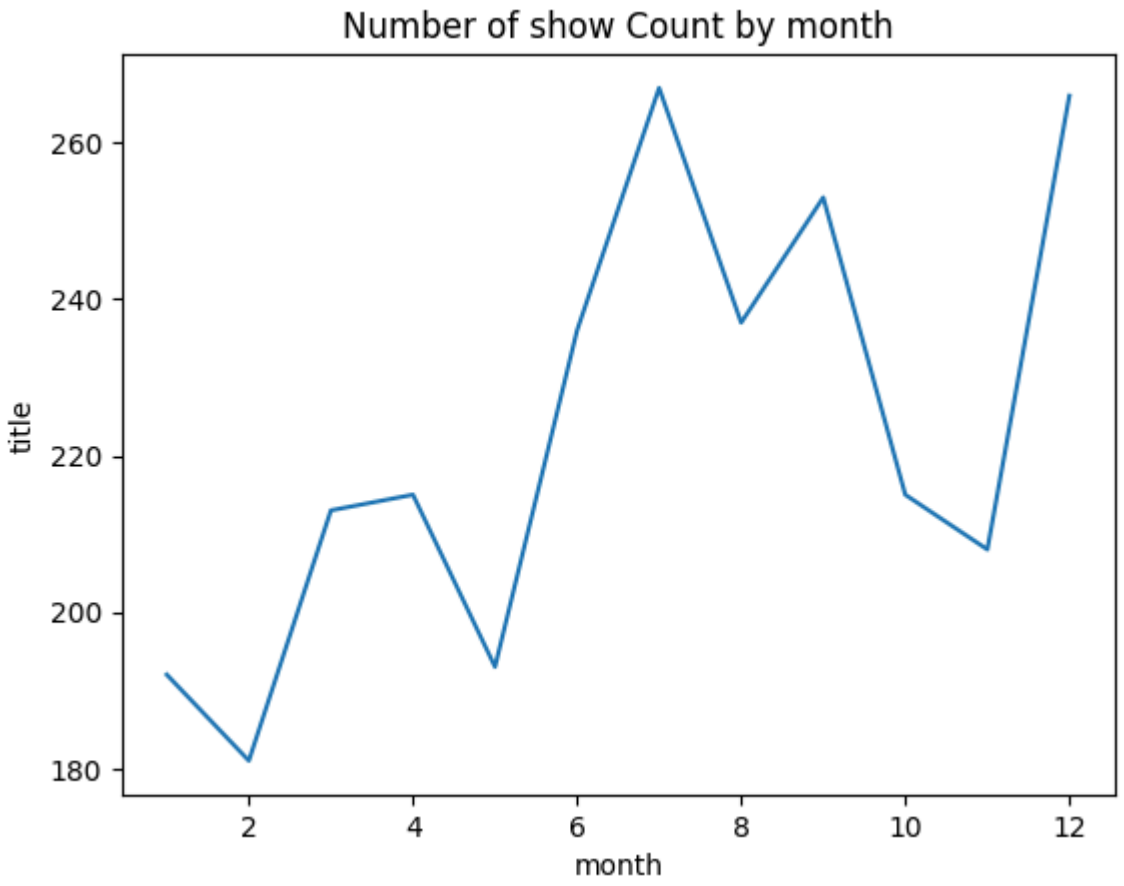


In [137…
```
dfs_month = df_show.groupby("month").agg({"title":"nunique"}).reset_index().sort_va
dfs_month.head(3)
```

Out[137…

	month	title
6	7	267
11	12	266
8	9	253

In [138…
```
sns.lineplot(x="month",y="title",data=dfs_month)
plt.title("Number of show Count by month")
plt.show()
```

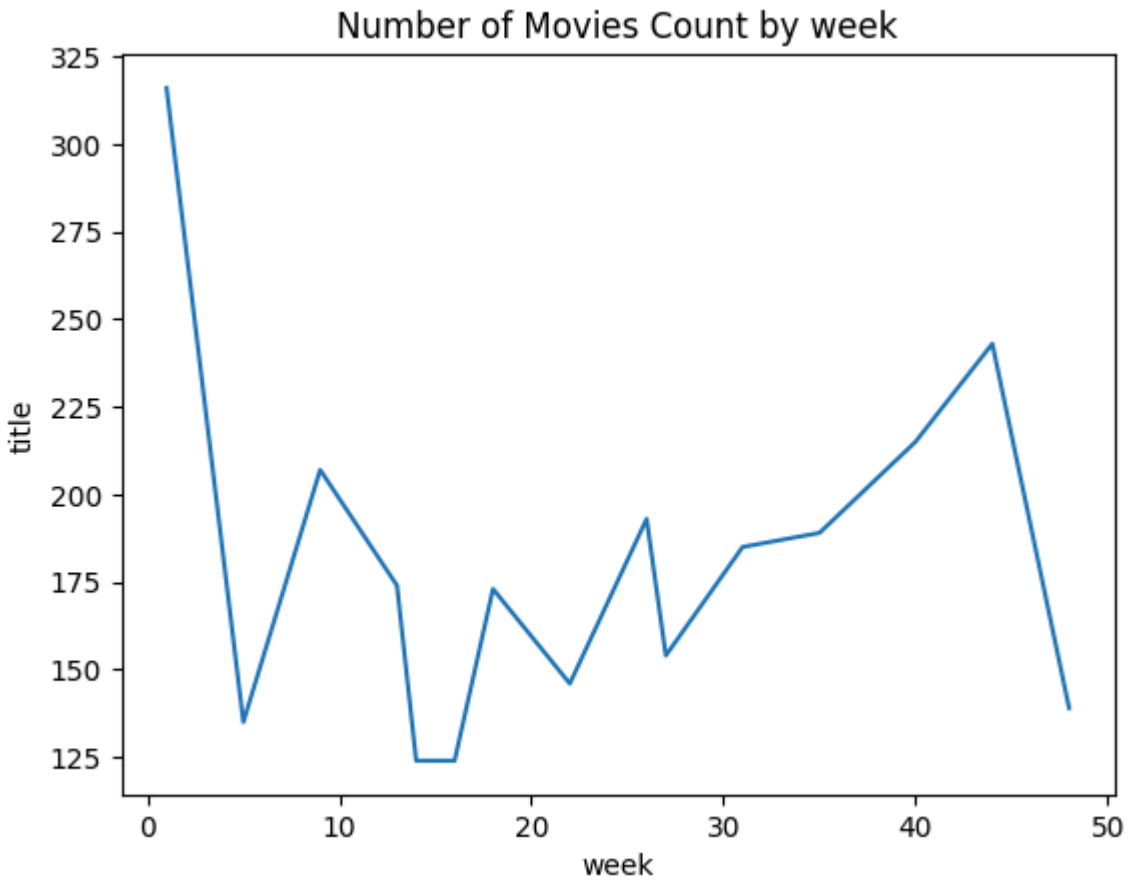


In [139…
```
dfm_week = df_movie.groupby("week").agg({"title":"nunique"}).reset_index().sort_val
dfm_week.head(3)
```

Out[139…

	week	title
0	1	316
43	44	243
39	40	215

In [140…
```
sns.lineplot(x="week",y="title",data=dfm_week)
plt.title("Number of Movies Count by week")
plt.show()
```

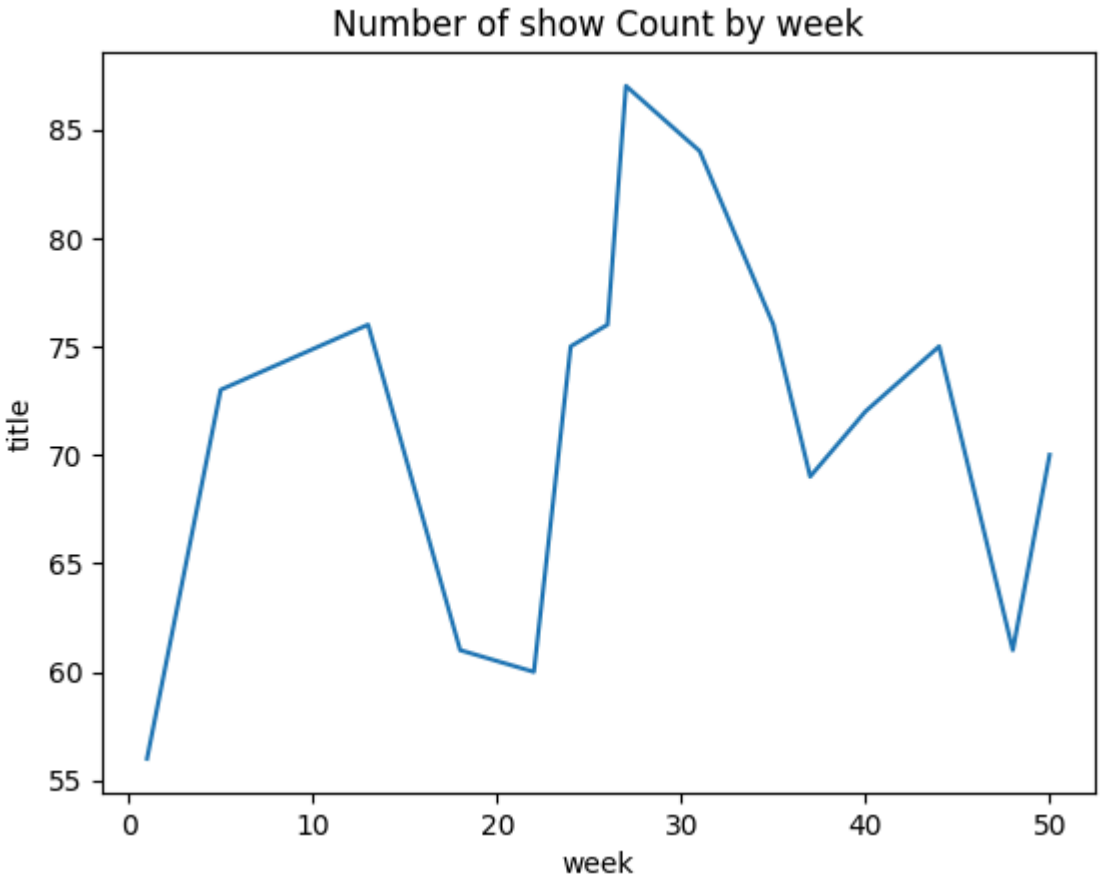


In [141…
```
dfs_week = df_show.groupby("week").agg({"title":"nunique"}).reset_index().sort_valu
dfs_week.head(3)
```

Out[141…

	week	title
26	27	87
30	31	84
34	35	76

In [142…
```
sns.lineplot(x="week",y="title",data=dfs_week)
plt.title("Number of show Count by week")
plt.show()
```



TV Shows are added in Netflix by a tremendous amount in mid weeks/months of the year, i.e- July

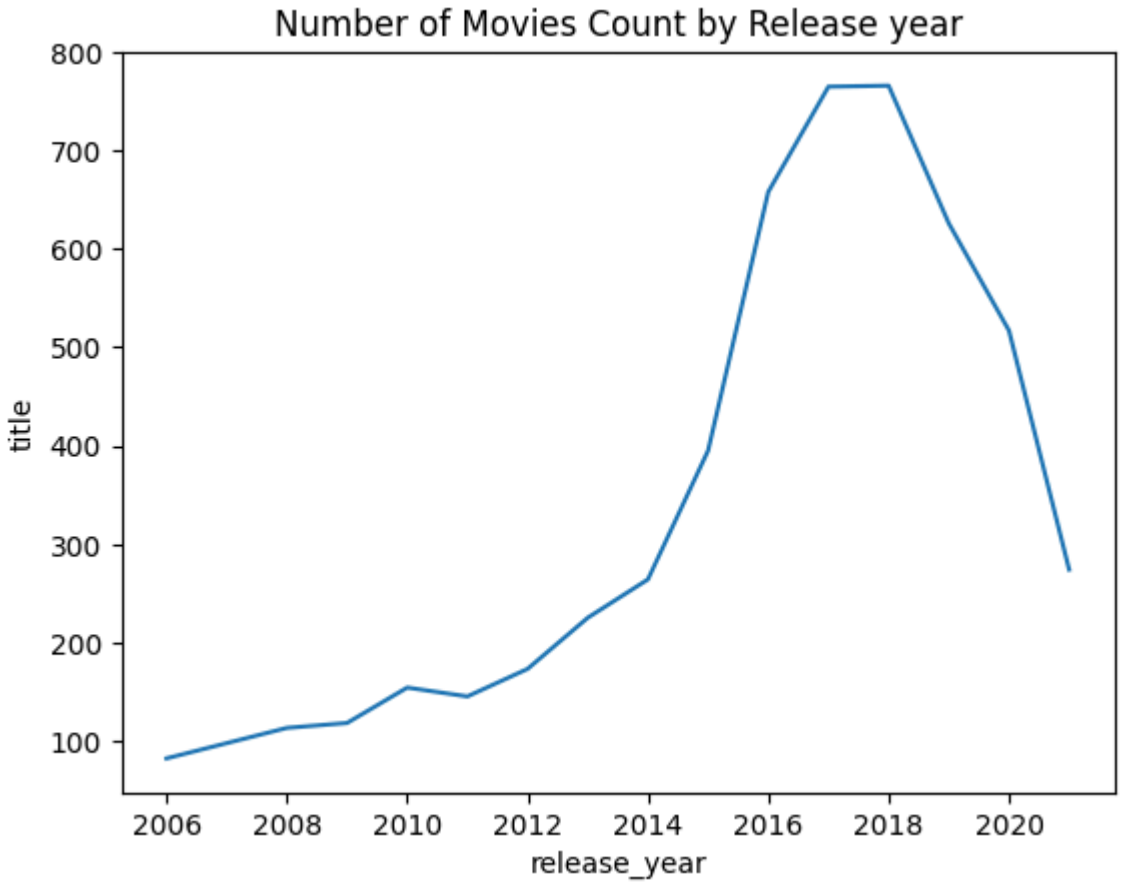
Movies are added in Netflix by a tremendous amount in first week/last month of current year and first month of next year

In [143…
```
dfm_release_year = df_movie.loc[df_movie["release_year"] >= 1980].groupby("release_
dfm_release_year.head(3)
```

Out[143…

	release_year	title
38	2018	766
37	2017	765
36	2016	658

In [144…
```
sns.lineplot(x="release_year",y="title",data=dfm_release_year)
plt.title("Number of Movies Count by Release year")
plt.show()
```

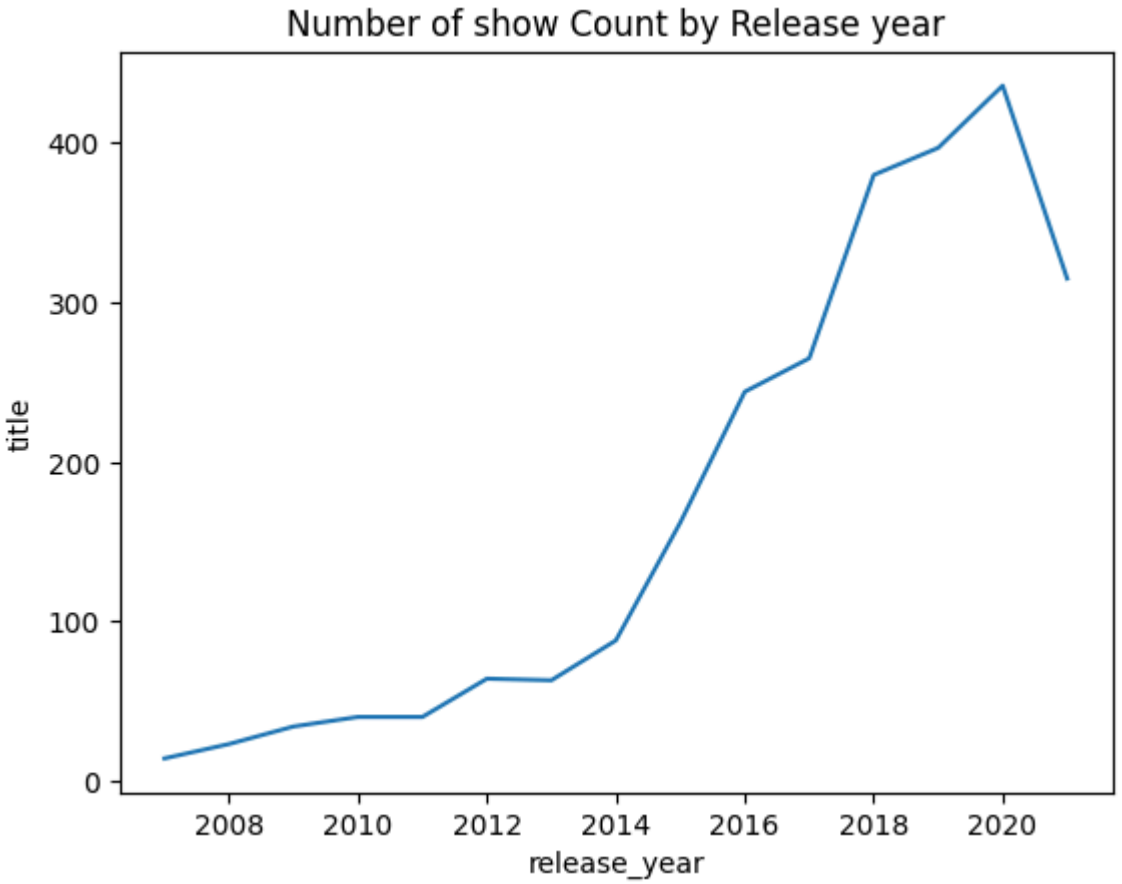


In [145…
```
dfs_release_year = df_show.loc[df_show["release_year"] >= 1980].groupby("release_ye
dfs_release_year.head(3)
```

Out[145…

	release_year	title
35	2020	436
34	2019	397
33	2018	380

In [146…
```
sns.lineplot(x="release_year",y="title",data=dfs_release_year)
plt.title("Number of show Count by Release year")
plt.show()
```



Actual Releases of both TV Shows and Movies have taken a hit after 2020

Univariate Analysis separately for shows and movies in USA

---

1. So this time, the granularity level is country and analysis of TV Shows/Movies the country brings. I am going to consider only the top countries individually for TV Shows and Movies. There are definitely some common countries too which bring out quality content in both TV Shows and Movies.
2. Which Genres do these countries offer and what are the intended audiences(Ratings) which are popular in Netflix?

3)In case of Movies, what is the duration/length of movies which makes them special and depicts attention span?

4)Who are the popular actors/directors across TV Shows and Movies in these countries?

5)In what time of the year, people tend to watch movies and shows in these countries?

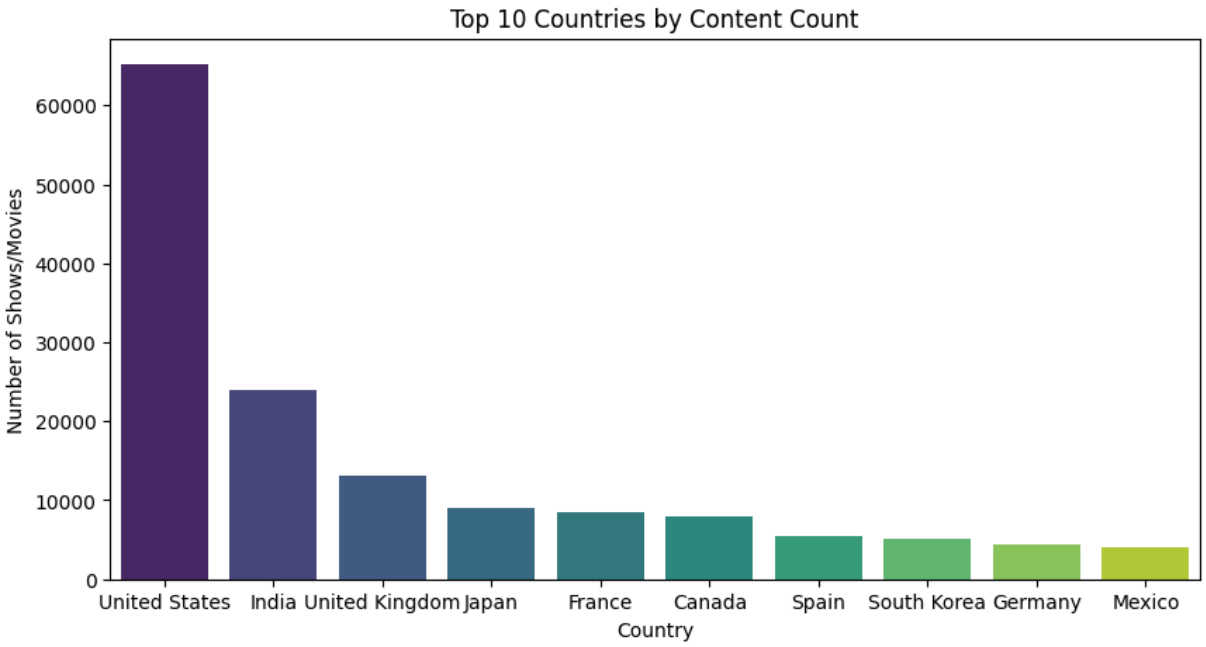
6)Popular Actor and Director Combinations in these countries

In [147…

```
country_content = df_final3['country'].value_counts().head(10)
plt.figure(figsize=(10,5))
```

```
sns.barplot(x=country_content.index, y=country_content.values, palette='viridis')
plt.title('Top 10 Countries by Content Count')
plt.ylabel('Number of Shows/Movies')
plt.xlabel('Country')
plt.show()
```

```
C:\Users\ARJUN\AppData\Local\Temp\ipykernel_20096\4099928481.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=country_content.index, y=country_content.values, palette='viridis')
```



In [148…
```
######
country_content_show =df_final.groupby(["type","country"]).size().reset_index(name=
top_show_country = country_content_show[country_content_show["type"]=="TV Show"][1:
top_show_country
```

Out[148…

	type	country	count
150	TV Show	Japan	5164
172	TV Show	South Korea	3757

In [149…
```
# below countries will be analyzed for both shows and movies
shows_and_movies=['United States','India']
```

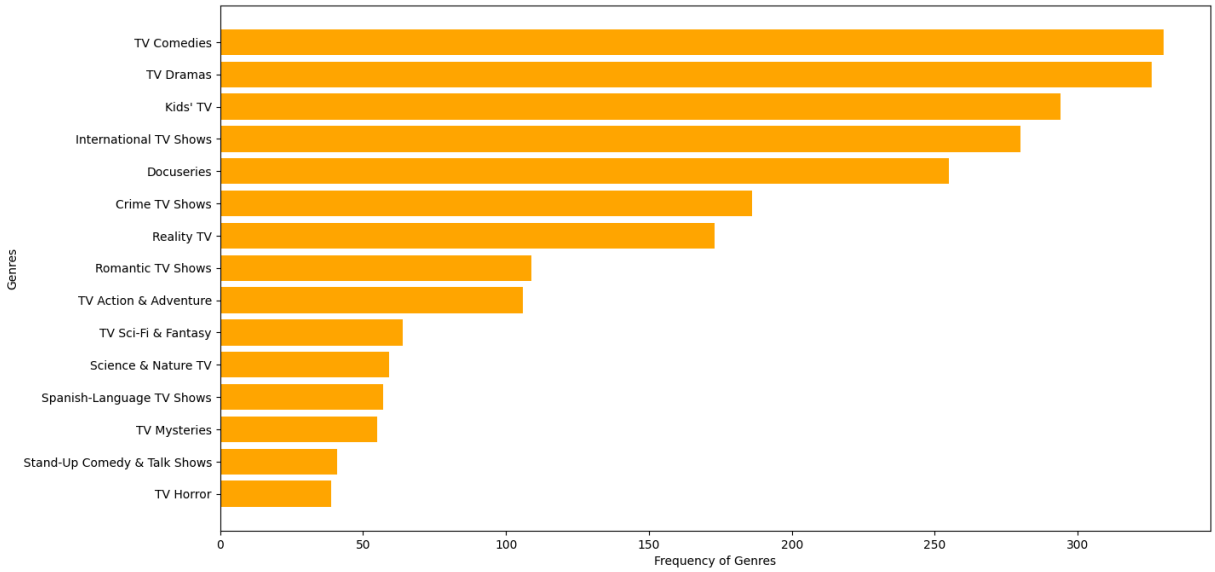
Univariate Analysis separately for shows and movies in USA

In [150…
```
#Analyzing USA for both shows and movies
```

In [151…
```
df_usa_movies = df_final3.loc[(df_final3["country"].str.contains("United States"))
df_usa_shows = df_final3.loc[(df_final3["country"].str.contains("United States")) &
```

In [152…

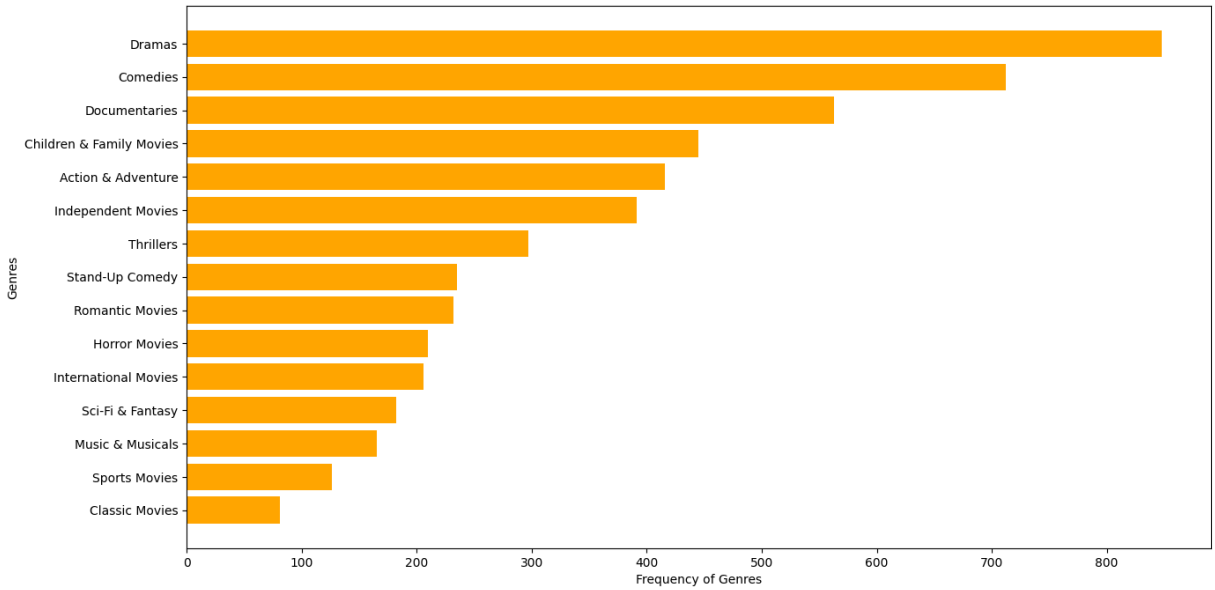
```
df_genre=df_usa_shows.groupby(['Genre']).agg({"title":"nunique"}).reset_index().sor
plt.figure(figsize=(15,8))
plt.barh(df_genre[::-1]['Genre'], df_genre[::-1]['title'],color=['orange'])
plt.xlabel('Frequency of Genres')
plt.ylabel('Genres')
plt.show()
```



Dramas,Comedy, Kids 'TV Shows, International TV Shows and Docuseries, Genres are popular in TV Series in USA

In [153…

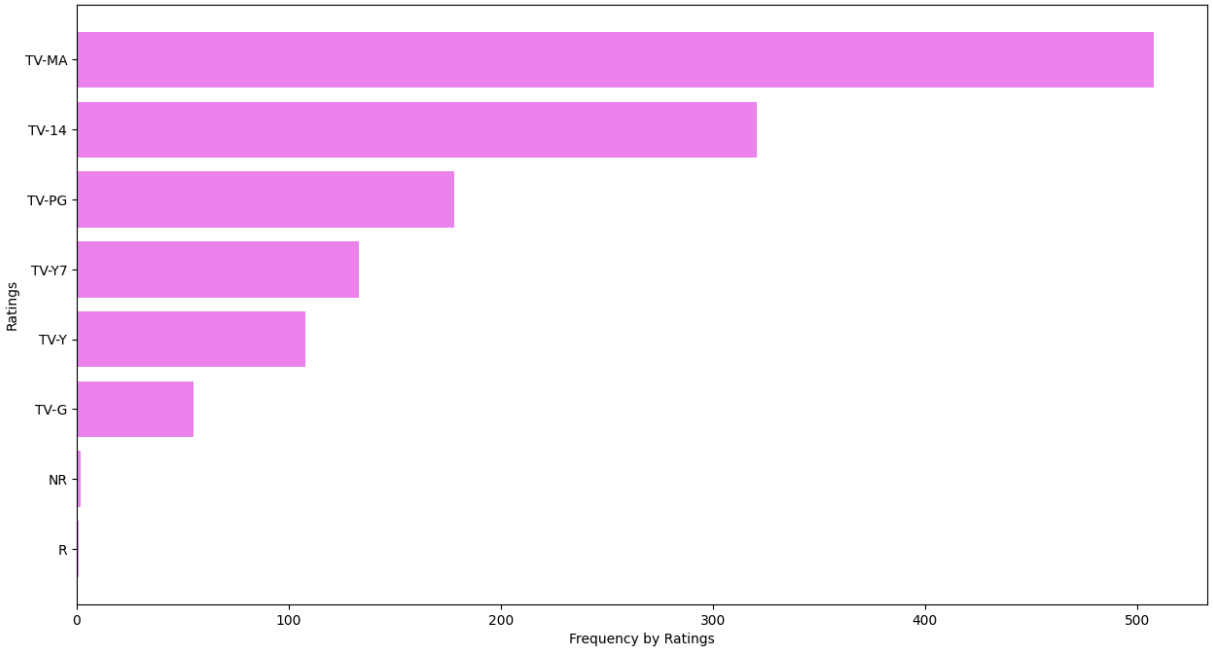
```
df_genre=df_usa_movies.groupby(['Genre']).agg({"title":"nunique"}).reset_index().so
plt.figure(figsize=(15,8))
plt.barh(df_genre[::-1]['Genre'], df_genre[::-1]['title'],color=['orange'])
plt.xlabel('Frequency of Genres')
plt.ylabel('Genres')
plt.show()
```



Dramas,Comedy, Documentaries, Family Movies and Action Genres in Movies are popular in USA

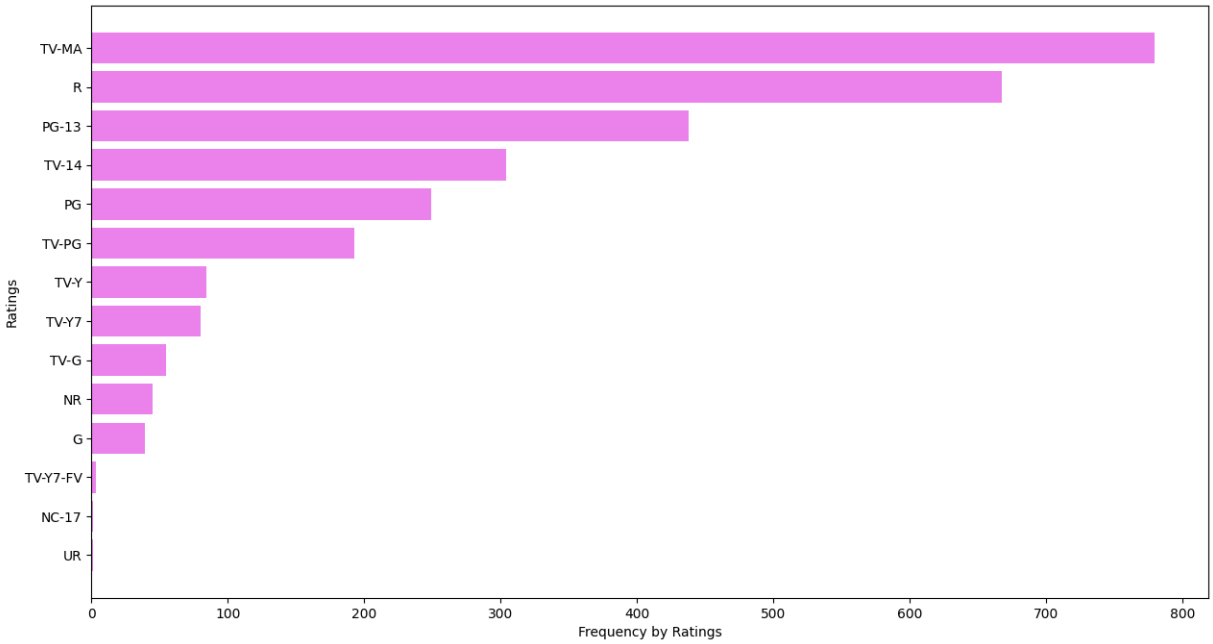
In [154…

```
df_rating=df_usa_shows.groupby(['rating']).agg({"title":"nunique"}).reset_index().s
plt.figure(figsize=(15,8))
plt.barh(df_rating[::-1]['rating'], df_rating[::-1]['title'],color=['violet'])
plt.xlabel('Frequency by Ratings')
plt.ylabel('Ratings')
plt.show()
```



In [155…

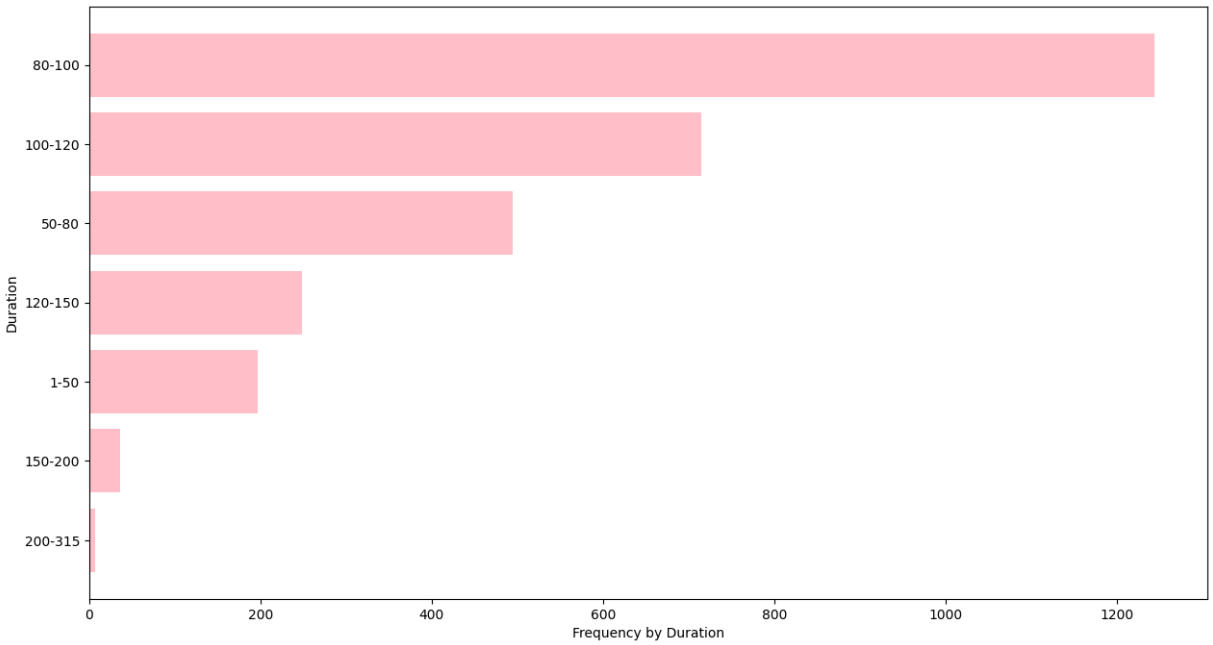
```
df_rating=df_usa_movies.groupby(['rating']).agg({"title":"nunique"}).reset_index().
plt.figure(figsize=(15,8))
plt.barh(df_rating[::-1]['rating'], df_rating[::-1]['title'],color=['violet'])
plt.xlabel('Frequency by Ratings')
plt.ylabel('Ratings')
plt.show()
```



So it seems plaussible to conclude that the popular ratings across Netflix includes Mature Audiences and those appropriate for over 14/over 17 ages in both Movies and TV Shows in USA

In [156…

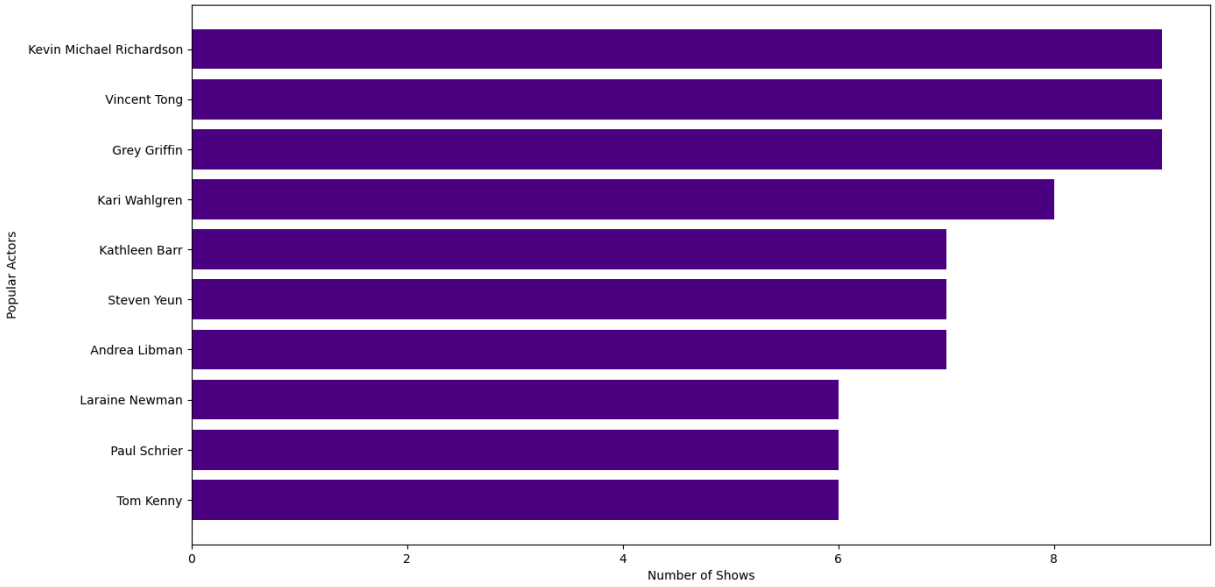
```
df_duration=df_usa_movies.groupby(['duration']).agg({"title":"nunique"}).reset_inde
plt.figure(figsize=(15,8))
plt.barh(df_duration[::-1]['duration'], df_duration[::-1]['title'],color=['pink'])
plt.xlabel('Frequency by Duration')
plt.ylabel('Duration')
plt.show()
```



Across movies 80-100,100-120 is the ranges of minutes for which most movies lie. So quite possibly 80-120 mins is the sweet spot we would be wanting for movies in USA

In [159…

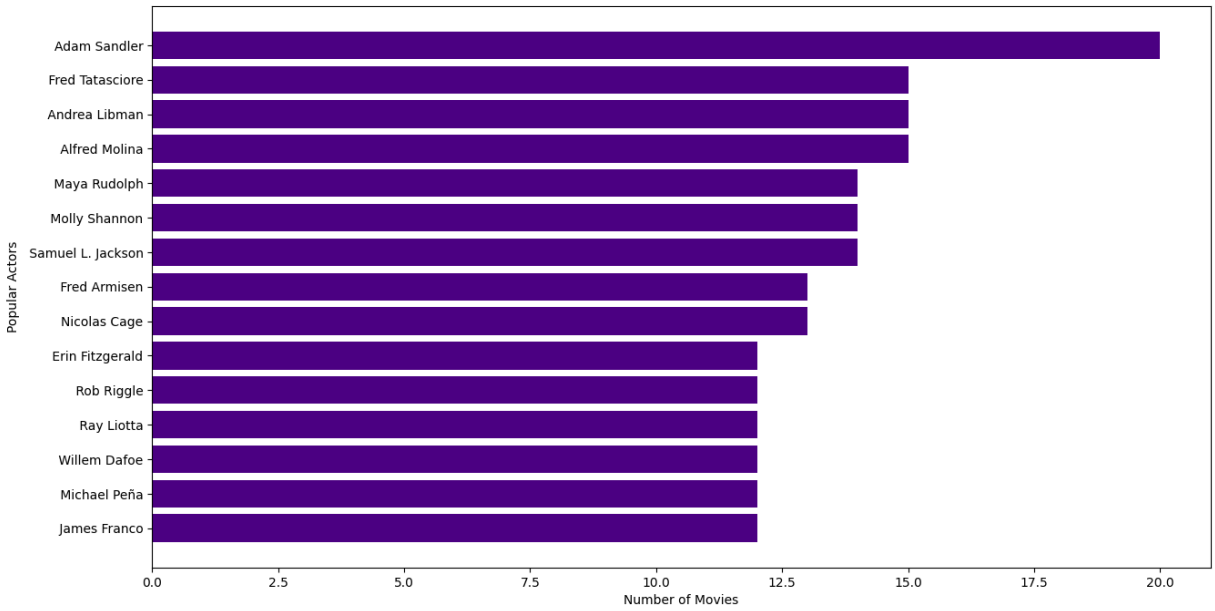
```
df_actors=df_usa_shows.loc[df_usa_shows["Actors"]!="Unknow Actors"].groupby(['Actor
plt.figure(figsize=(15,8))
plt.barh(df_actors[::-1]['Actors'], df_actors[::-1]['title'],color=['indigo'])
plt.xlabel('Number of Shows')
plt.ylabel('Popular Actors')
plt.show()
```



Vincent Tong,Grey Griffin and Kevin Richardson are the most popular actors across TV Shows in USA

In [161…

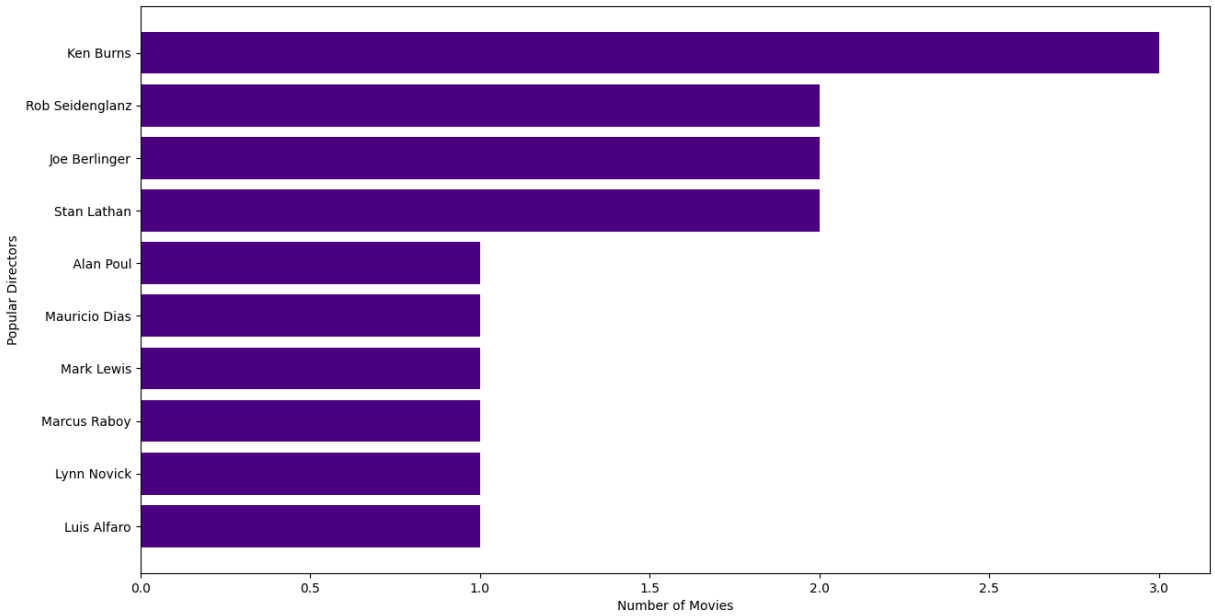
```
df_actors=df_usa_movies.loc[df_usa_movies["Actors"]!="Unknow Actors"].groupby(['Act
plt.figure(figsize=(15,8))
plt.barh(df_actors[::-1]['Actors'], df_actors[::-1]['title'],color=['indigo'])
plt.xlabel('Number of Movies')
plt.ylabel('Popular Actors')
plt.show()
```



Samuel Jackson,Adam Sandler,James Franco and Nicolas Cage are very much popular across movies on Netflix in USA

In [162…

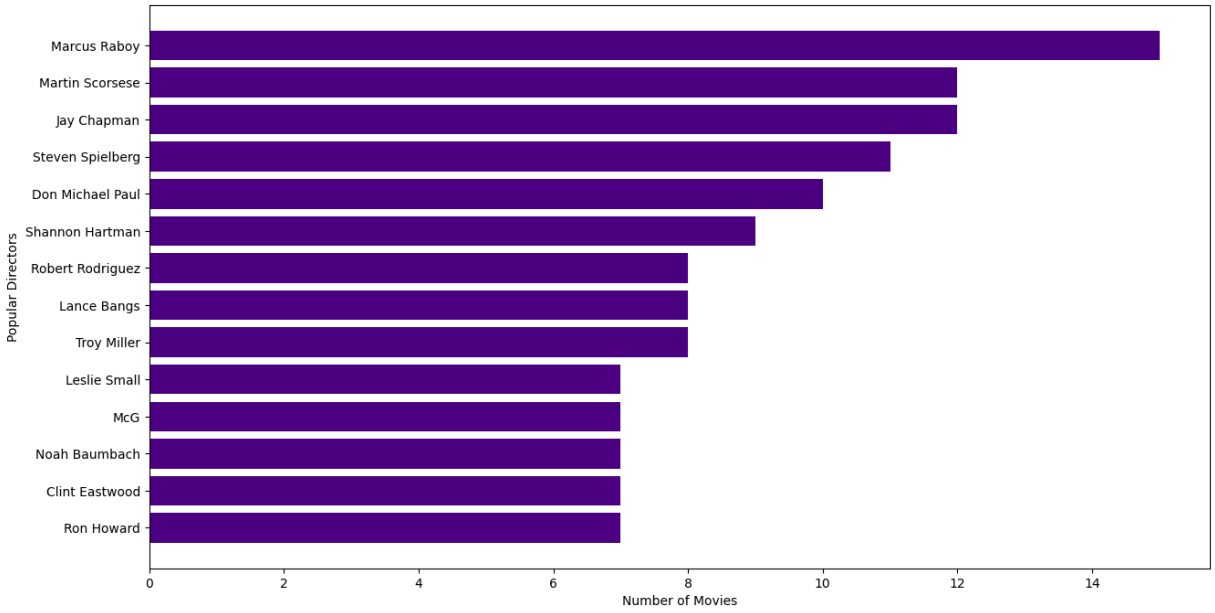
```
df_directors=df_usa_shows.loc[df_usa_shows["Director"]!="Unknow Director"].groupby(
plt.figure(figsize=(15,8))
plt.barh(df_directors[::-1]['Director'], df_directors[::-1]['title'],color=['indigo
plt.xlabel('Number of Movies')
plt.ylabel('Popular Directors')
plt.show()
```



Ken Burns,Stan Lathan, Joe Barlinger are popular directors across TV Shows on Netflix in USA

In [163…

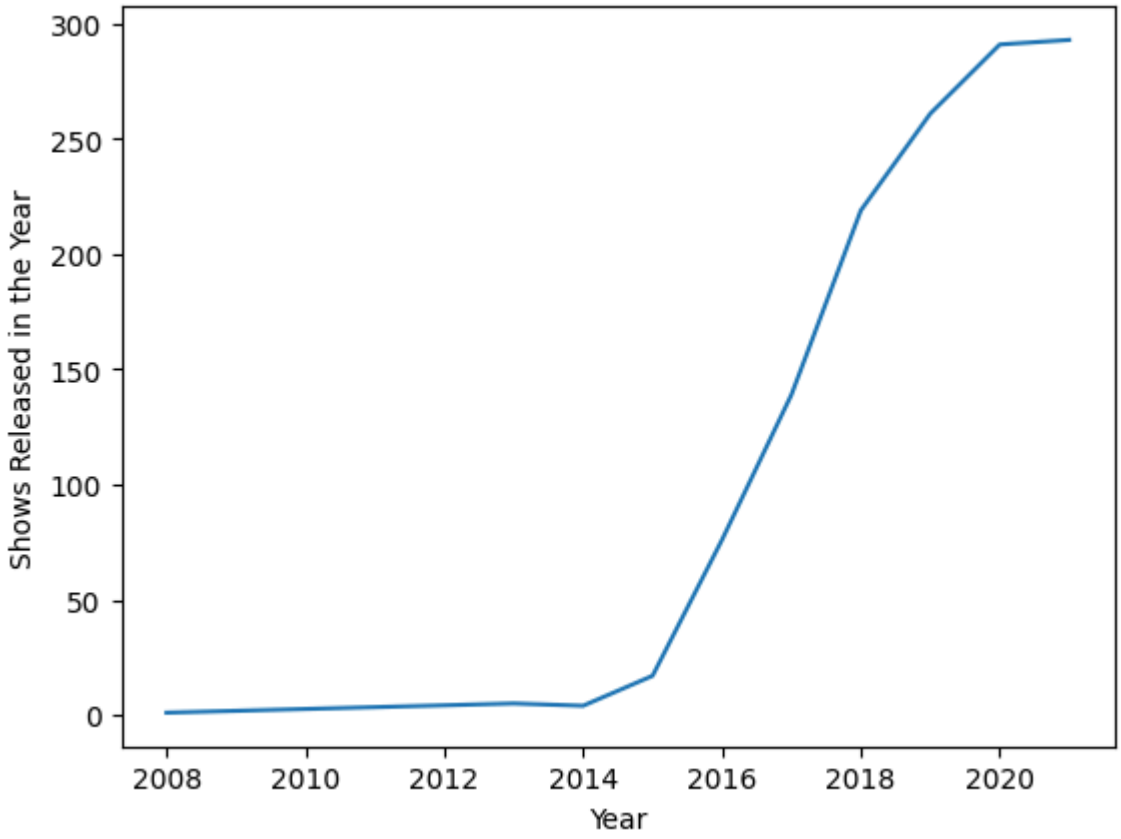
```
df_directors=df_usa_movies.loc[df_usa_movies["Director"]!="Unknow Director"].groupb
plt.figure(figsize=(15,8))
plt.barh(df_directors[::-1]['Director'], df_directors[::-1]['title'],color=['indigo
plt.xlabel('Number of Movies')
plt.ylabel('Popular Directors')
plt.show()
```



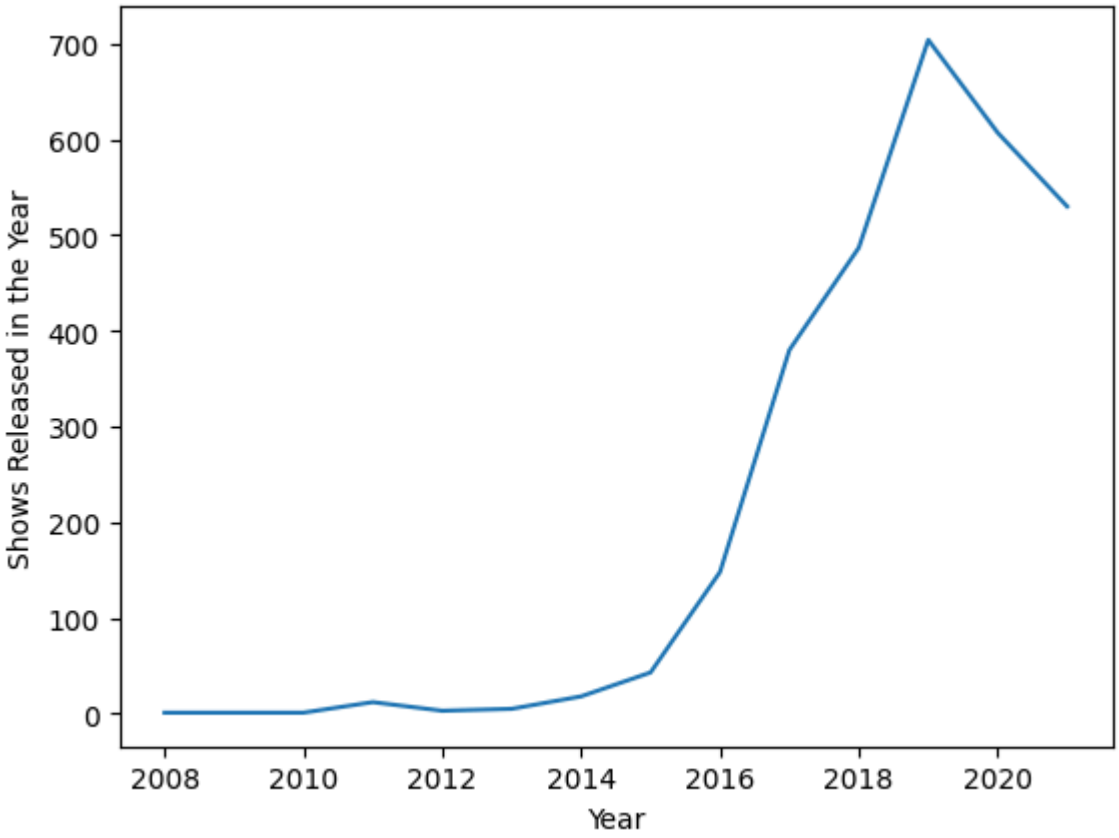
Jay Karas,Marcus Raboy,Martin Scorcese and Jay Chapman are popular directors across movies in USA

In [164…

```
df_year=df_usa_shows.groupby(['year']).agg({"title":"nunique"}).reset_index()
sns.lineplot(data=df_year, x='year', y='title')
plt.ylabel("Shows Released in the Year")
plt.xlabel("Year")
plt.show()
```

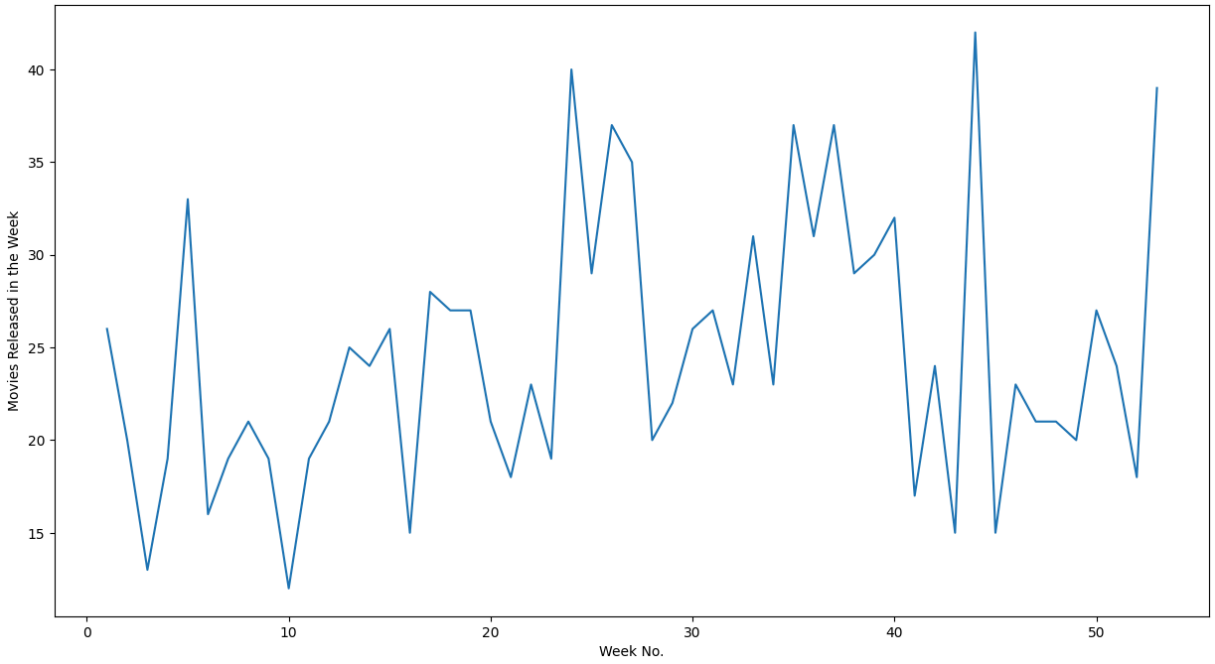


In [165…
```
df_year=df_usa_movies.groupby(['year']).agg({"title":"nunique"}).reset_index()
sns.lineplot(data=df_year, x='year', y='title')
plt.ylabel("Shows Released in the Year")
plt.xlabel("Year")
plt.show()
```



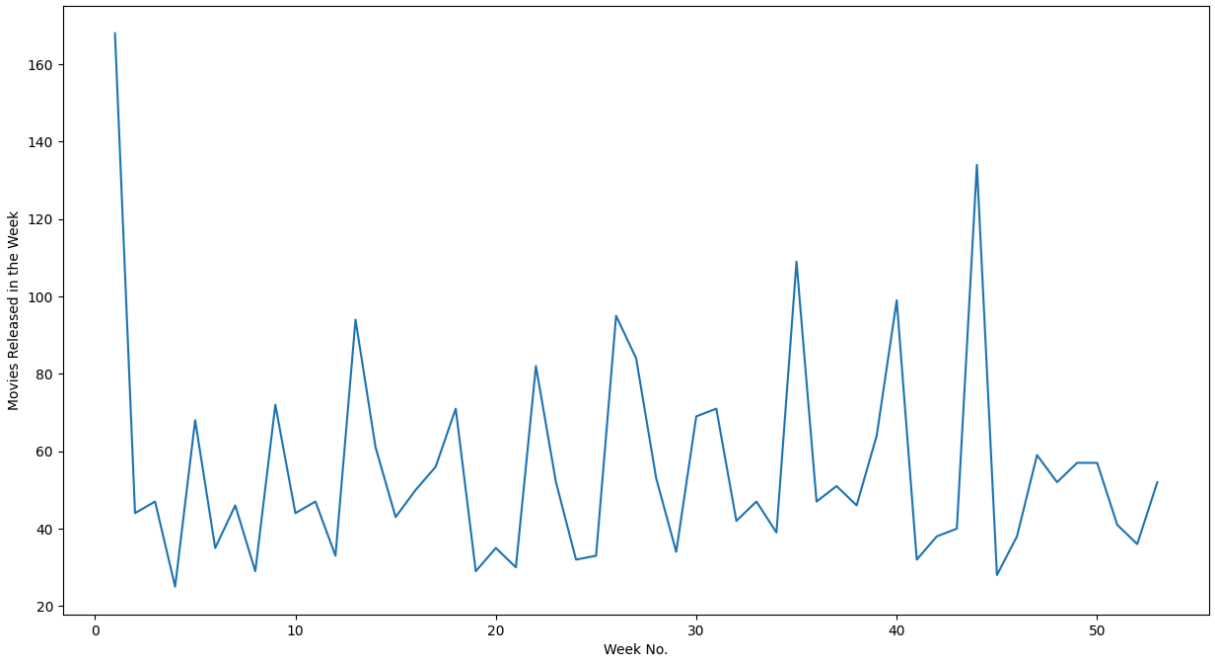
In USA, number of shows remained the same in 2021 as they were in 2020 while number of movies declined:

In [166…
```
df_week=df_usa_shows.groupby(['week']).agg({"title":"nunique"}).reset_index()
plt.figure(figsize=(15,8))
sns.lineplot(data=df_week, x='week', y='title')
plt.ylabel("Movies Released in the Week")
plt.xlabel("Week No.")
plt.show()
```



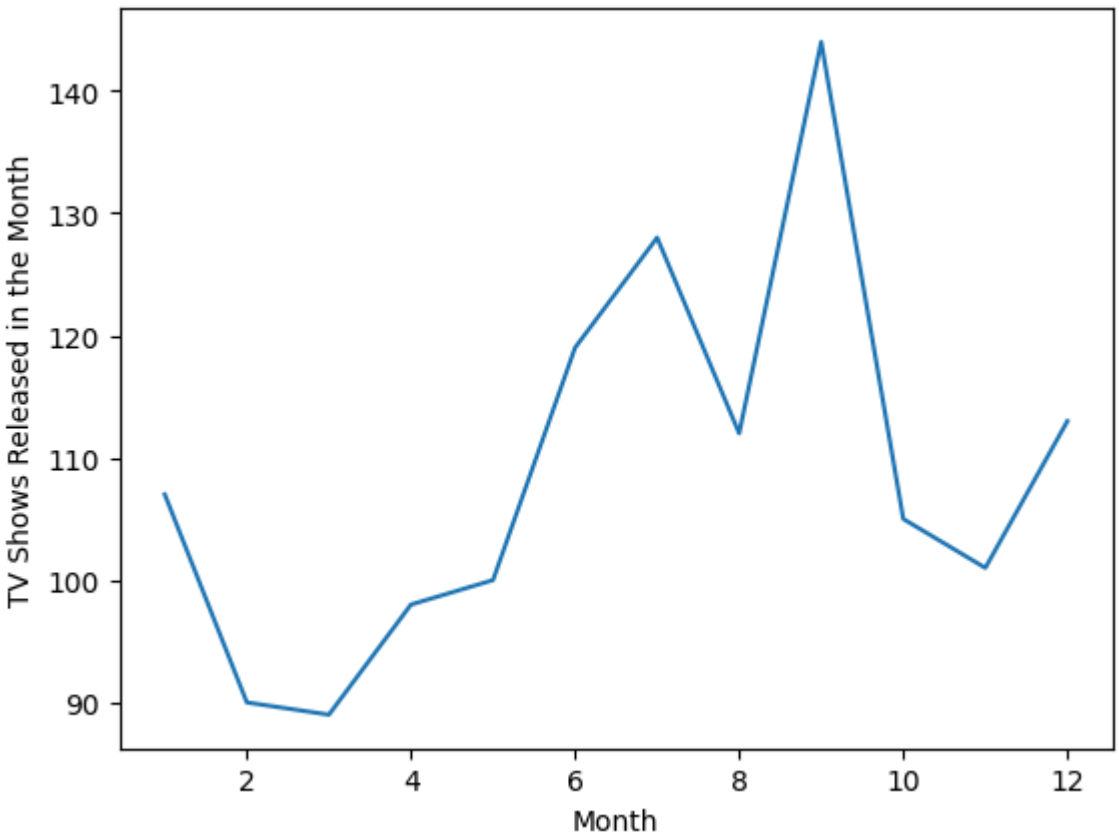
In [167…

```
df_week=df_usa_movies.groupby(['week']).agg({"title":"nunique"}).reset_index()
plt.figure(figsize=(15,8))
sns.lineplot(data=df_week, x='week', y='title')
plt.ylabel("Movies Released in the Week")
plt.xlabel("Week No.")
plt.show()
```



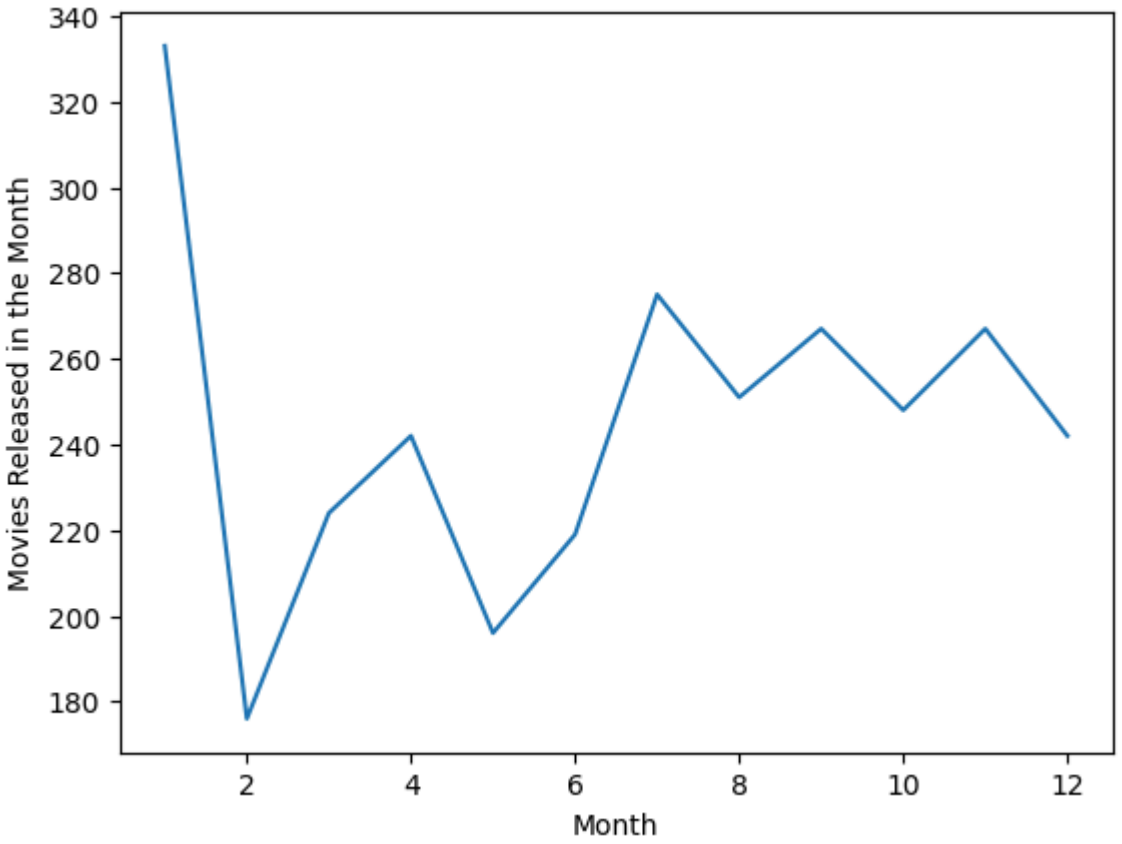
In [168…

```
df_month=df_usa_shows.groupby(['month']).agg({"title":"nunique"}).reset_index()
sns.lineplot(data=df_month, x='month', y='title')
plt.ylabel("TV Shows Released in the Month")
plt.xlabel("Month")
plt.show()
```



In [169…

```
df_month=df_usa_movies.groupby(['month']).agg({"title":"nunique"}).reset_index()
sns.lineplot(data=df_month, x='month', y='title')
plt.ylabel("Movies Released in the Month")
plt.xlabel("Month")
plt.show()
```

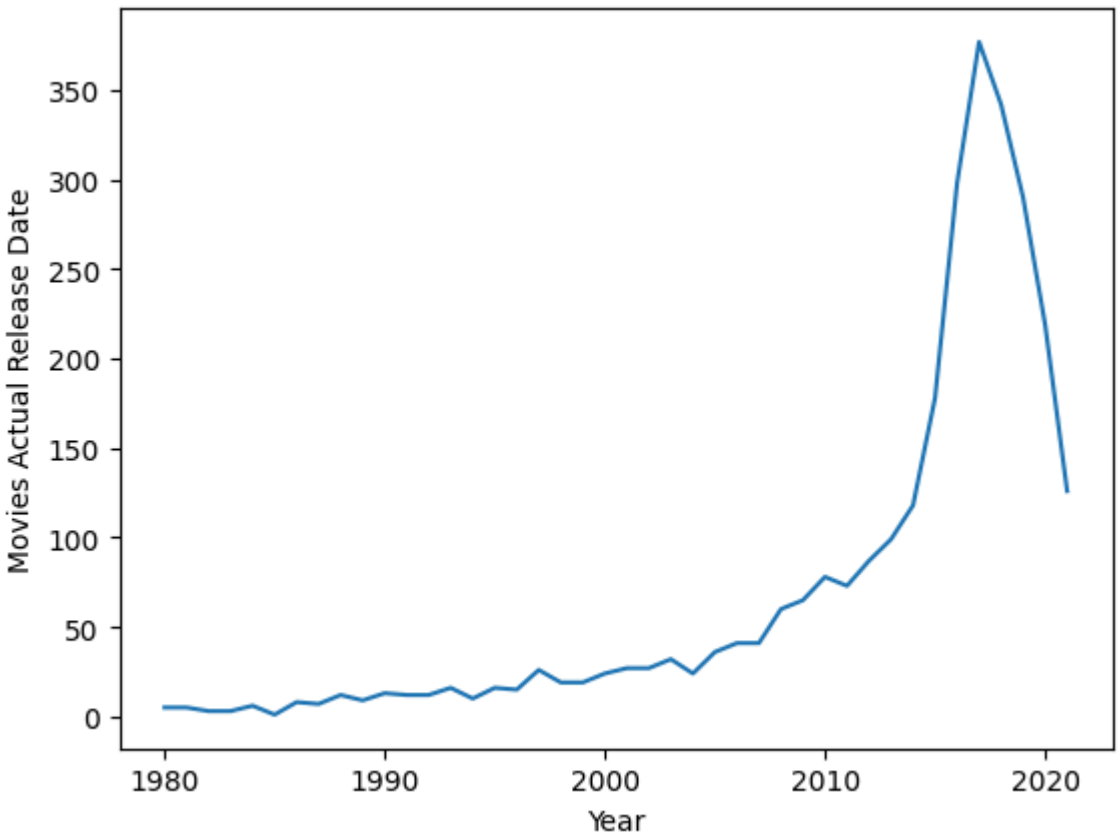


TV Shows are added in Netflix by a tremendous amount in July and September in USA

Movies are added in Netflix in USA by a tremendous amount in first week/last month of current year and first month of next year

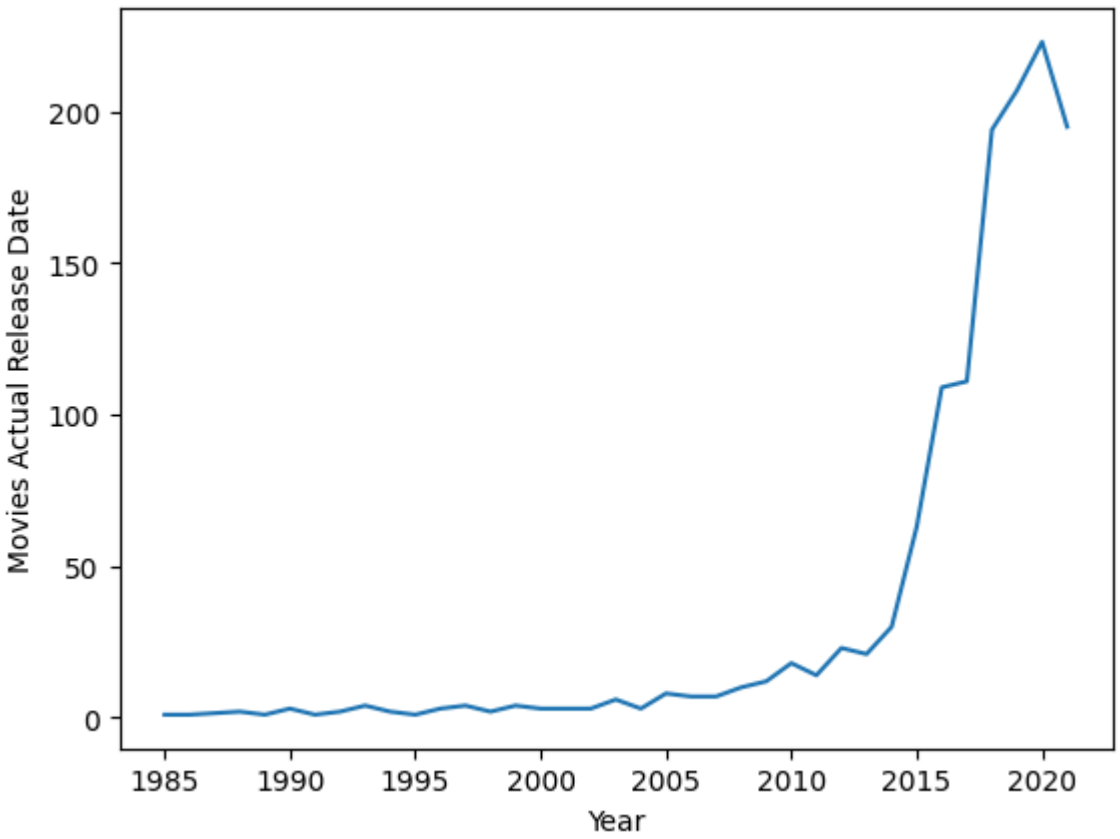
In [170…

```
df_release_year=df_usa_movies[df_usa_movies['release_year']>=1980].groupby(['releas
sns.lineplot(data=df_release_year, x='release_year', y='title')
plt.ylabel("Movies Actual Release Date")
plt.xlabel("Year")
plt.show()
```



In [171…

```
df_release_year=df_usa_shows[df_usa_shows['release_year']>=1980].groupby(['release_
sns.lineplot(data=df_release_year, x='release_year', y='title')
plt.ylabel("Movies Actual Release Date")
plt.xlabel("Year")
plt.show()
```



In USA, though both Movies and Shows have reduced in 2021, the amount of decrease in number of TV Shows is small as compared to Movies

In [172…

```
#Analysing a combination of actors and directors
df_usa_movies["Actor_Director_Combination"] = df_usa_movies["Actors"].str.cat(df_us
df_usa_movies_subset = df_usa_movies.loc[~df_usa_movies["Actors"].str.contains("Unk
df_usa_movies_subset = df_usa_movies_subset.loc[~df_usa_movies_subset["Director"].s
df_usa_movies_subset.head(5)
```

```
C:\Users\ARJUN\AppData\Local\Temp\ipykernel_20096\522472293.py:2: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  df_usa_movies["Actor_Director_Combination"] = df_usa_movies["Actors"].str.cat(df_u
sa_movies["Director"],sep=" and ")
```

Out[172…

	title	Actors	Director	Genre	country	show_id	type	date_added	releas
159	My Little Pony: A New Generation	Vanessa Hudgens	Robert Cullen	Children & Family Movies	United States	s7	Movie	September 24, 2021	
160	My Little Pony: A New Generation	Vanessa Hudgens	José Luis Ucha	Children & Family Movies	United States	s7	Movie	September 24, 2021	
161	My Little Pony: A New Generation	Kimiko Glenn	Robert Cullen	Children & Family Movies	United States	s7	Movie	September 24, 2021	
162	My Little Pony: A New Generation	Kimiko Glenn	José Luis Ucha	Children & Family Movies	United States	s7	Movie	September 24, 2021	
163	My Little Pony: A New Generation	James Marsden	Robert Cullen	Children & Family Movies	United States	s7	Movie	September 24, 2021	

In [173…

```
df_usa_shows['Actor_Director_Combination'] = df_usa_shows.Actors.str.cat(df_usa_sho
df_usa_shows_subset=df_usa_shows.loc[df_usa_shows['Actors'] != 'Unknow Actors']
df_usa_shows_subset=df_usa_shows_subset.loc[df_usa_shows_subset['Director']!='Unkno
df_usa_shows_subset.head()
```

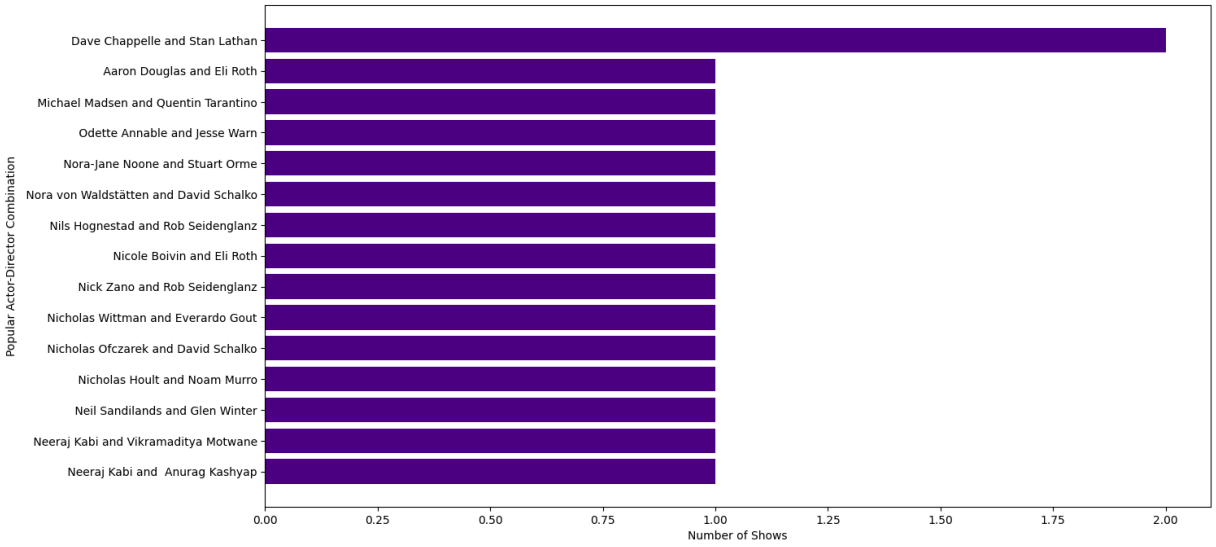
```
C:\Users\ARJUN\AppData\Local\Temp\ipykernel_20096\2376891389.py:1: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  df_usa_shows['Actor_Director_Combination'] = df_usa_shows.Actors.str.cat(df_usa_sh
ows.Director, sep=' and ')
```

Out[173...

	title	Actors	Director	Genre	country	show_id	type	date_added	release_y
111	Midnight Mass	Kate Siegel	Mike Flanagan	TV Dramas	United States	s6	TV Show	September 24, 2021	2(
112	Midnight Mass	Kate Siegel	Mike Flanagan	TV Horror	United States	s6	TV Show	September 24, 2021	2(
113	Midnight Mass	Kate Siegel	Mike Flanagan	TV Mysteries	United States	s6	TV Show	September 24, 2021	2(
114	Midnight Mass	Zach Gilford	Mike Flanagan	TV Dramas	United States	s6	TV Show	September 24, 2021	2(
115	Midnight Mass	Zach Gilford	Mike Flanagan	TV Horror	United States	s6	TV Show	September 24, 2021	2(

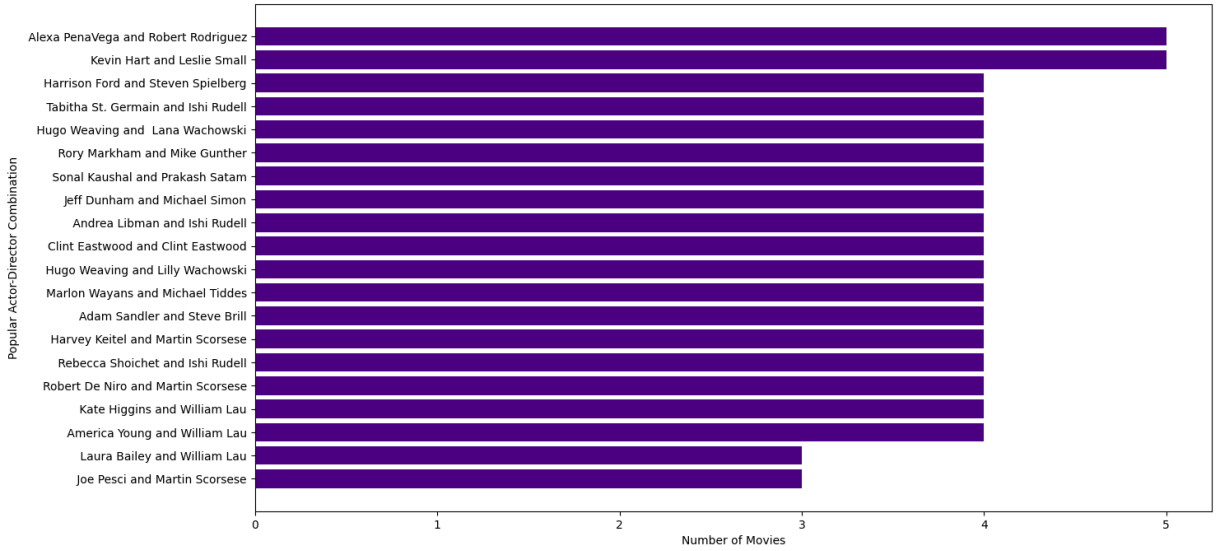
In [174...

```
df_actors_directors=df_usa_shows_subset.groupby(['Actor_Director_Combination']).agg
plt.figure(figsize=(15,8))
plt.barh(df_actors_directors[::-1]['Actor_Director_Combination'], df_actors_directo
plt.xlabel('Number of Shows')
plt.ylabel('Popular Actor-Director Combination')
plt.show()
```



In [175...

```
df_actors_directors=df_usa_movies_subset.groupby(['Actor_Director_Combination']).ag
plt.figure(figsize=(15,8))
plt.barh(df_actors_directors[::-1]['Actor_Director_Combination'], df_actors_directo
plt.xlabel('Number of Movies')
plt.ylabel('Popular Actor-Director Combination')
plt.show()
```



In [176…
```
df_actors_directors[::-1]['Actor_Director_Combination'].values
```

Out[176…
```
array([' Joe Pesci and Martin Scorsese', ' Laura Bailey and William Lau',
       ' America Young and William Lau', ' Kate Higgins and William Lau',
       'Robert De Niro and Martin Scorsese',
       ' Rebecca Shoichet and Ishi Rudell',
       ' Harvey Keitel and Martin Scorsese',
       'Adam Sandler and Steve Brill', 'Marlon Wayans and Michael Tiddes',
       ' Hugo Weaving and Lilly Wachowski',
       'Clint Eastwood and Clint Eastwood',
       ' Andrea Libman and Ishi Rudell', 'Jeff Dunham and Michael Simon',
       ' Sonal Kaushal and Prakash Satam',
       ' Rory Markham and Mike Gunther',
       ' Hugo Weaving and  Lana Wachowski',
       ' Tabitha St. Germain and Ishi Rudell',
       'Harrison Ford and Steven Spielberg',
       'Kevin Hart and Leslie Small',
       ' Alexa PenaVega and Robert Rodriguez'], dtype=object)
```

The Most Popular Actor Director Combination in Movies Across USA are:-

'Smith Foreman and Stanley Moore',
'Marlon Wayans and Michael Tiddes',
'Adam Sandler and Steve Brill',
'Maisie Benson and Stanley Moore',
'Ashleigh Ball and Ishi Rudell',
'Tara Strong and Ishi Rudell',
'Rebecca Shoichet and Ishi Rudell',
'Kerry Gudjohnsen and Alex Woo',
'Kerry Gudjohnsen and Stanley Moore',
'Paul Killam and Alex Woo',
'Paul Killam and Stanley Moore',
'Andrea Libman and Ishi Rudell',
'Kevin Hart and Leslie Small',

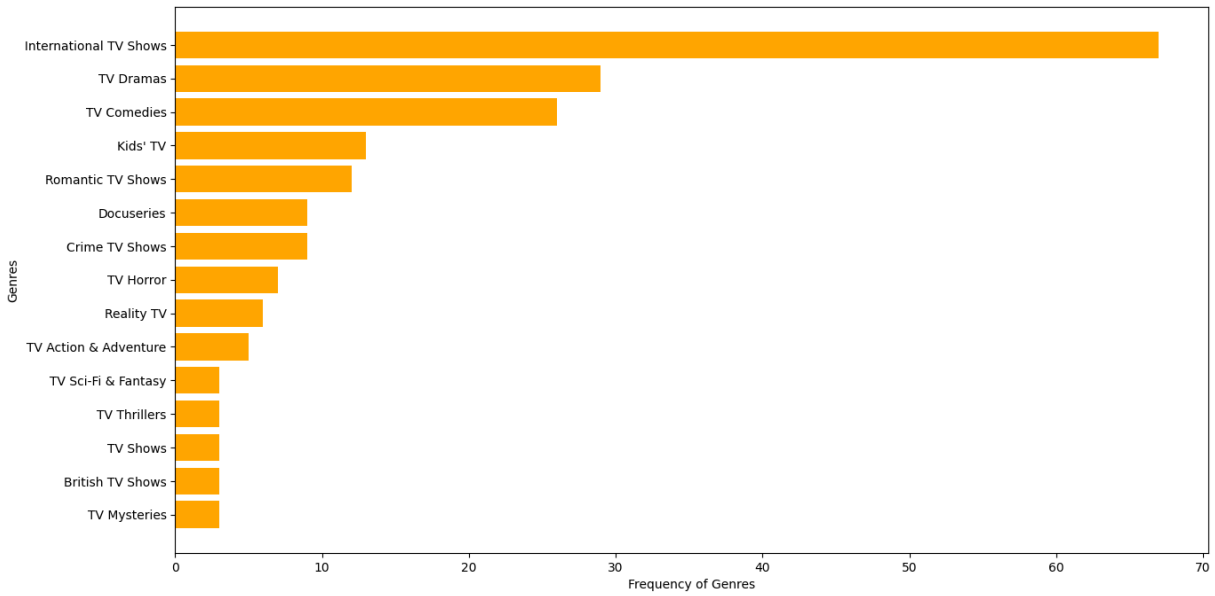
'Maisie Benson and Alex Woo',
'Alexa PenaVega and Robert Rodriguez',
'Tabitha St. Germain and Ishi Rudell'

The Second Most Popular Actor Director Combination in Movies Across USA are:-
'Rory Markham and Mike Gunther',
'Erin Mathews and Steve Ball',
'Danny Trejo and Robert Rodriguez',
'Jeff Dunham and Michael Simon'

Univariate Analysis separately for shows and movies in India

In [177…
```
#Analyzing India for both shows and movies

df_india_shows=df_final3.loc[ (df_final3['country']=='India') &  (df_final3['type']
df_india_movies=df_final3.loc[ (df_final3['country']=='India') & (df_final3['type']
```

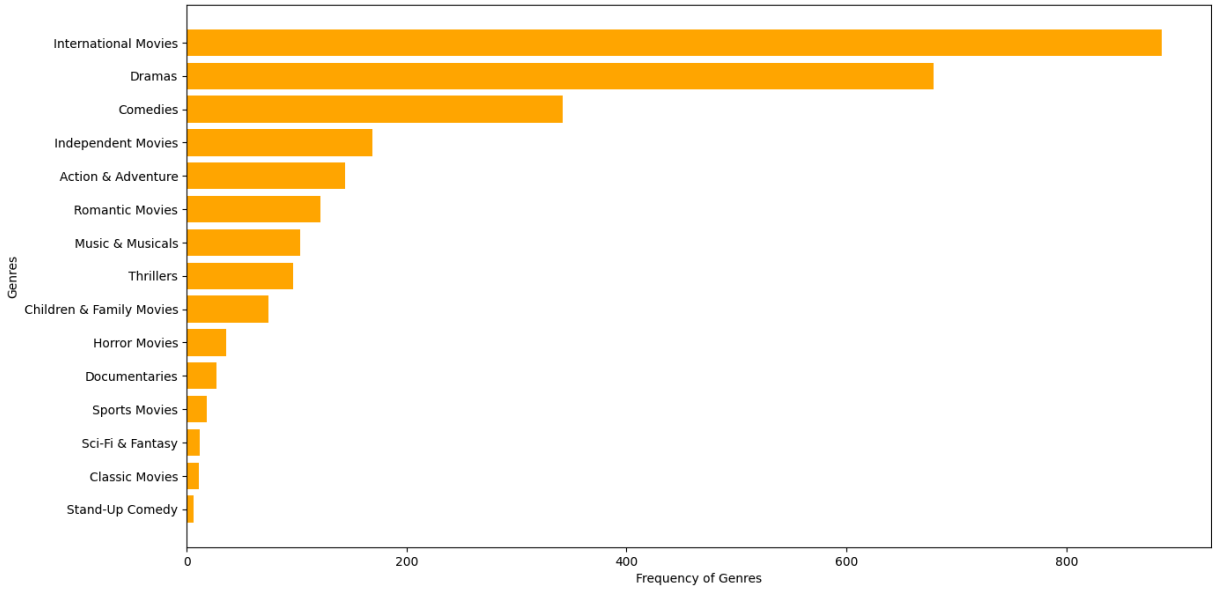
In [178…
```
df_genre=df_india_shows.groupby(['Genre']).agg({"title":"nunique"}).reset_index().s
plt.figure(figsize=(15,8))
plt.barh(df_genre[::-1]['Genre'], df_genre[::-1]['title'],color=['orange'])
plt.xlabel('Frequency of Genres')
plt.ylabel('Genres')
plt.show()
```



Dramas,Comedy, Kids 'TV Shows and International TV Shows Genres are popular in TV Series in India

In [179…
```
df_genre=df_india_movies.groupby(['Genre']).agg({"title":"nunique"}).reset_index().
plt.figure(figsize=(15,8))
plt.barh(df_genre[::-1]['Genre'], df_genre[::-1]['title'],color=['orange'])
plt.xlabel('Frequency of Genres')
```

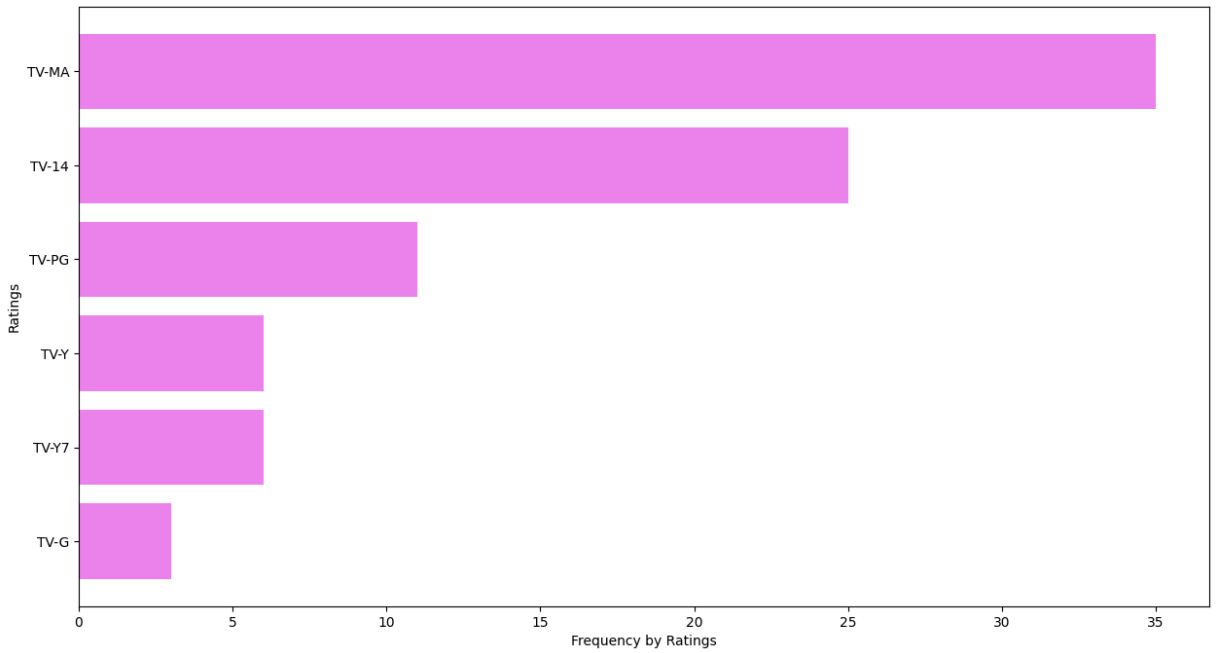
```
plt.ylabel('Genres')
plt.show()
```



International Movies,Drama,Comedy,Indpeendent Movies and Action, Romance Genres are prevalent in India

In [180…

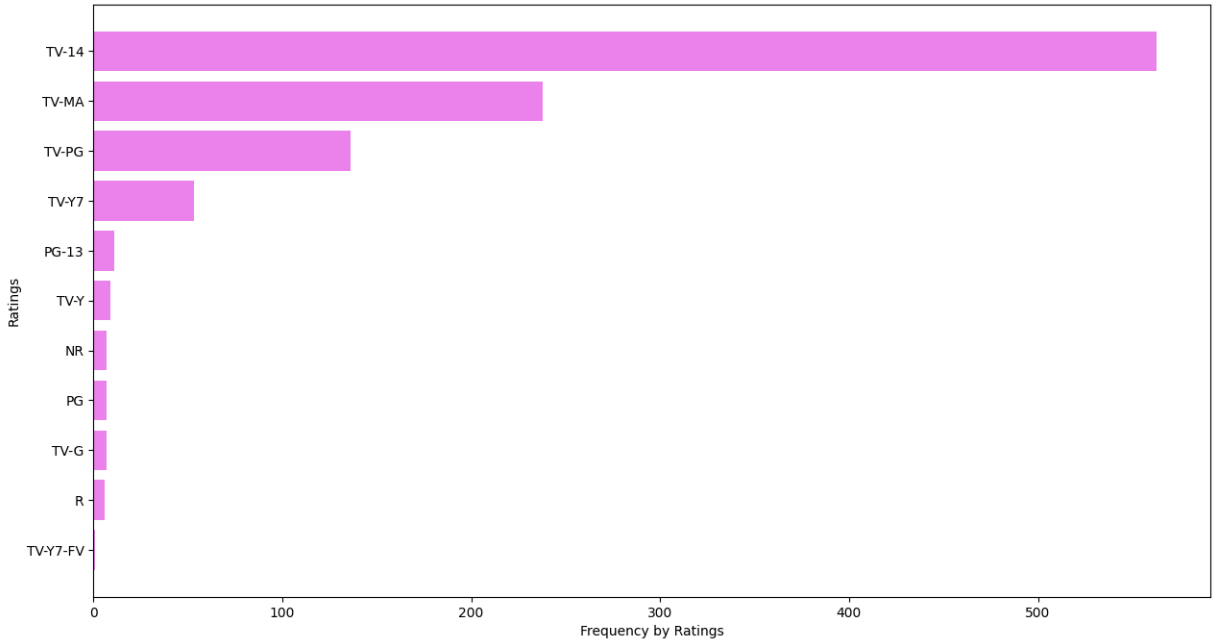
```
df_rating=df_india_shows.groupby(['rating']).agg({"title":"nunique"}).reset_index()
plt.figure(figsize=(15,8))
plt.barh(df_rating[::-1]['rating'], df_rating[::-1]['title'],color=['violet'])
plt.xlabel('Frequency by Ratings')
plt.ylabel('Ratings')
plt.show()
```



In [181…

```
df_rating=df_india_movies.groupby(['rating']).agg({"title":"nunique"}).reset_index(
plt.figure(figsize=(15,8))
plt.barh(df_rating[::-1]['rating'], df_rating[::-1]['title'],color=['violet'])
plt.xlabel('Frequency by Ratings')
```

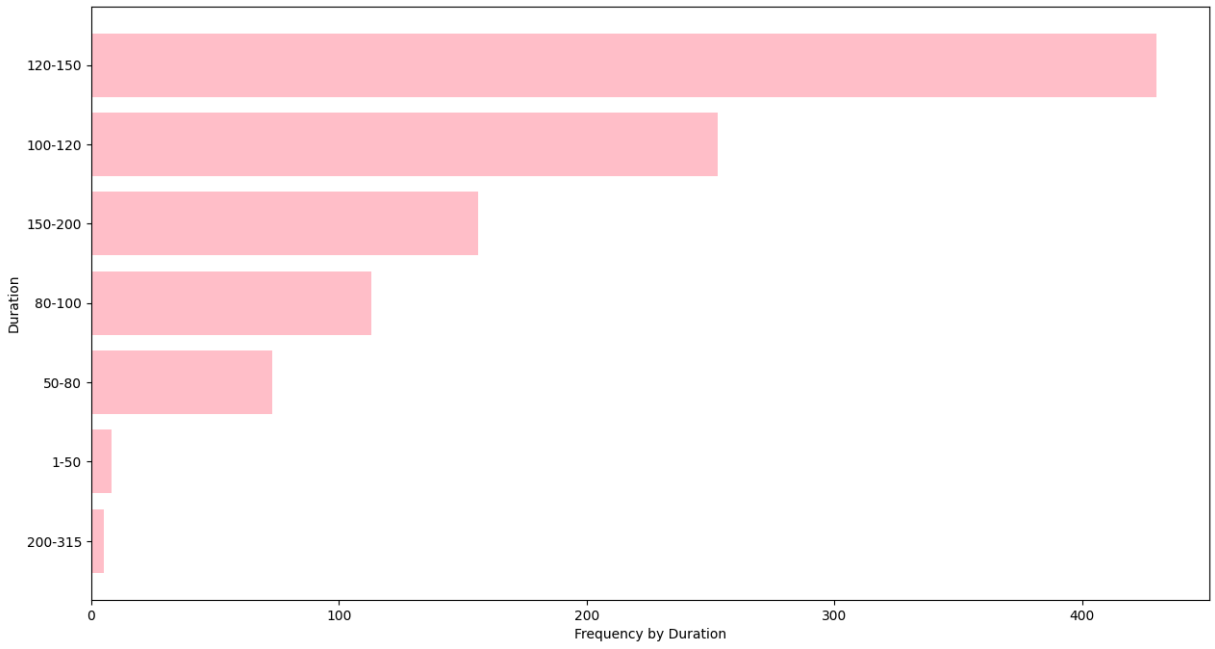
```
plt.ylabel('Ratings')
plt.show()
```



So it seems plaussible to conclude that the popular ratings across Netflix includes Mature Audiences in TV Shows and those appropriate for people over 14 in Movies in India.

Now this indeed seems to be the case. Indian TV Shows in Netflix are without a shadow of doubt intended for Mature Audiences while Movies for over 14 years of age.

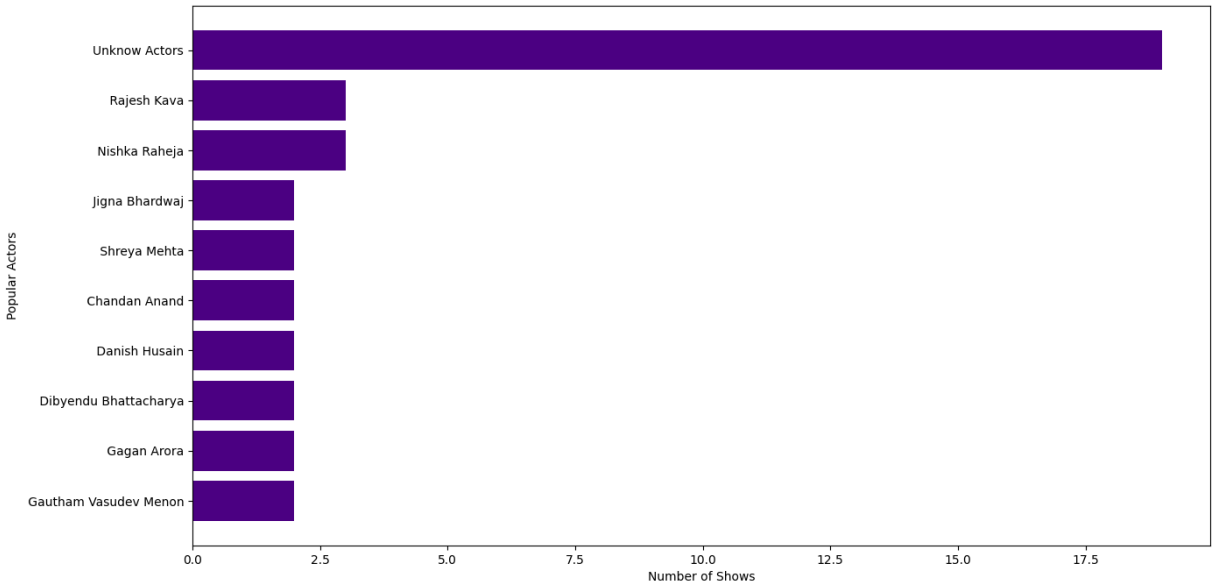
In [182…
```
df_duration=df_india_movies.groupby(['duration']).agg({"title":"nunique"}).reset_in
plt.figure(figsize=(15,8))
plt.barh(df_duration[::-1]['duration'], df_duration[::-1]['title'],color=['pink'])
plt.xlabel('Frequency by Duration')
plt.ylabel('Duration')
plt.show()
```



Across movies ranges of minutes in India are comparatively greater than USA with a sweet spot at 120-150 mins.

In [183…

```
df_actors=df_india_shows.groupby(['Actors']).agg({"title":"nunique"}).reset_index()
df_actors=df_actors[df_actors['Actors']!='Unknown Actor']
plt.figure(figsize=(15,8))
plt.barh(df_actors[::-1]['Actors'], df_actors[::-1]['title'],color=['indigo'])
plt.xlabel('Number of Shows')
plt.ylabel('Popular Actors')
plt.show()
```



Popular Actors in TV Shows in India are:-

'Rajesh Kava',

'Nishka Raheja',

'Prakash Raj',

'Sabina Malik',

'Anjali',

'Aranya Kaur',

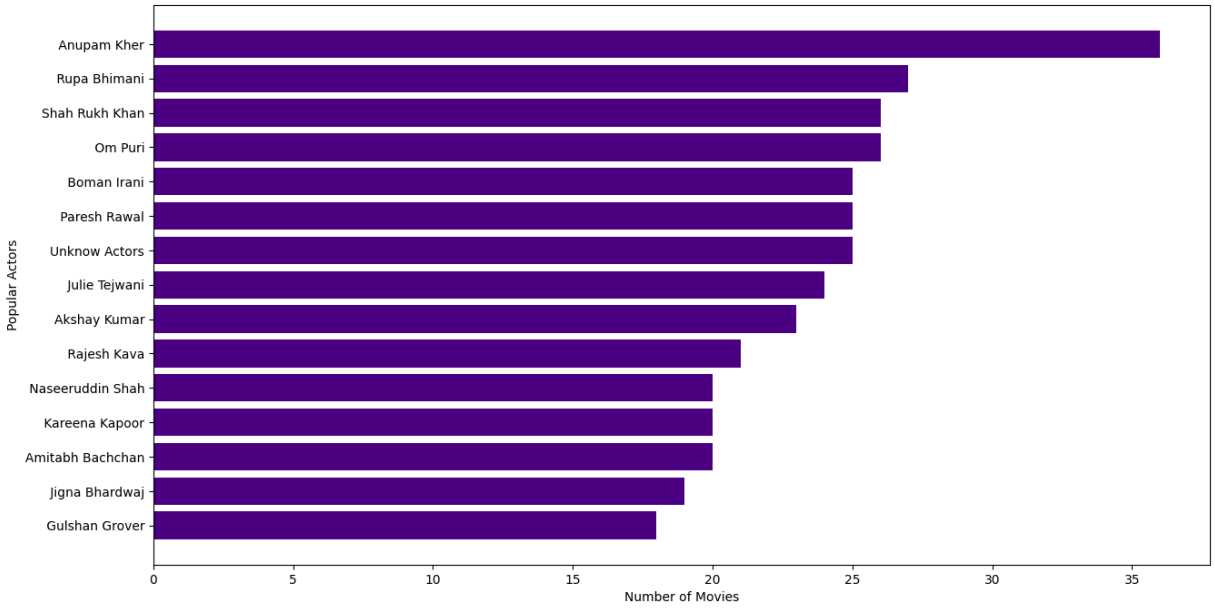
'Sonal Kaushal',

'Chandan Anand',

'Danish Husain'

In [184…

```
df_actors=df_india_movies.groupby(['Actors']).agg({"title":"nunique"}).reset_index(
df_actors=df_actors[df_actors['Actors']!='Unknown Actor']
plt.figure(figsize=(15,8))
plt.barh(df_actors[::-1]['Actors'], df_actors[::-1]['title'],color=['indigo'])
plt.xlabel('Number of Movies')
plt.ylabel('Popular Actors')
plt.show()
```

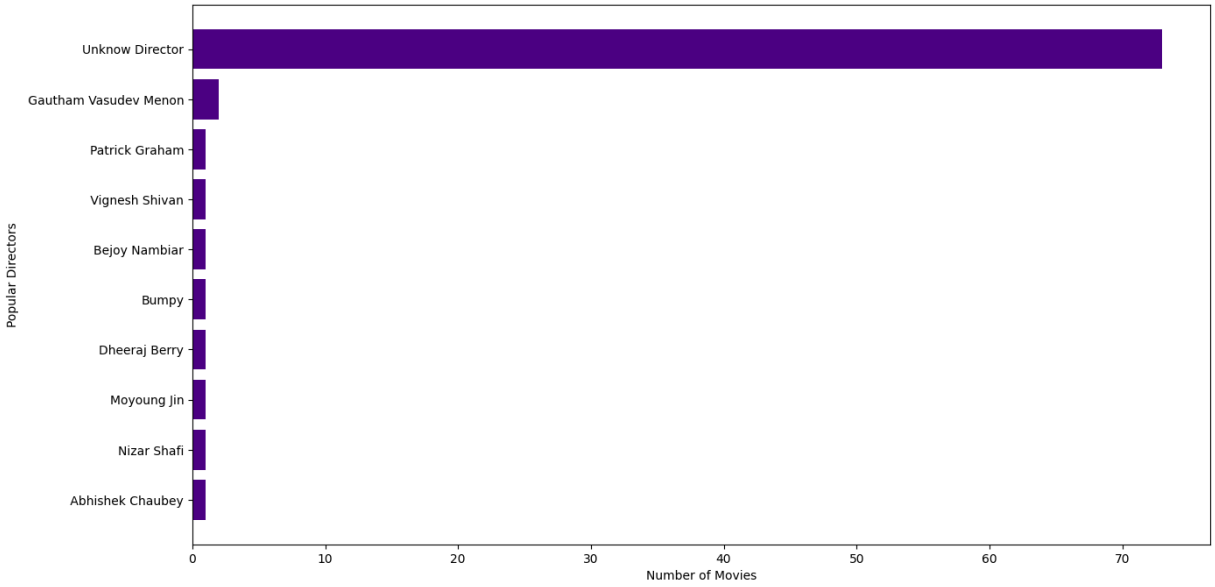


Popular actors across Movies in India:-

'Anupam Kher',
'Shah Rukh Khan',
'Naseeruddin Shah',
'Akshay Kumar',
'Om Puri',
'Paresh Rawal',
'Julie Tejwani',
'Amitabh Bachchan',
'Boman Irani',
'Rupa Bhimani',
'Kareena Kapoor',
'Ajay Devgn',
'Rajesh Kava',
'Kay Kay Menon'

In [185…

```
df_directors=df_india_shows.groupby(['Director']).agg({"title":"nunique"}).reset_in
df_directors=df_directors[df_directors['Director']!='Unknown Director']
plt.figure(figsize=(15,8))
plt.barh(df_directors[::-1]['Director'], df_directors[::-1]['title'],color=['indigo
plt.xlabel('Number of Movies')
plt.ylabel('Popular Directors')
plt.show()
```



Popular Directors Across Movies in India:-

'Gautham Vasudev Menon',

'Abhishek Chaubey',

'Sudha Kongara',

'Rathindran R Prasad',

'Sankalp Reddy',

'Sarjun',

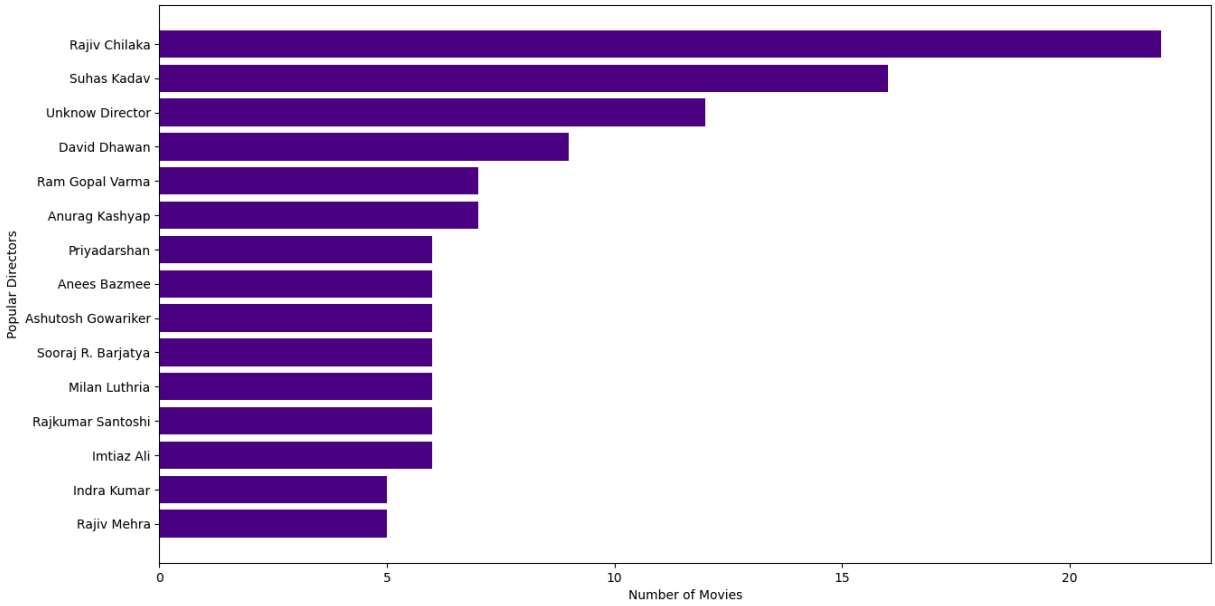
'Soumendra Padhi',

'Srijit Mukherji',

'Tharun Bhascker Dhaassyam'

In [186…

```
df_directors=df_india_movies.groupby(['Director']).agg({"title":"nunique"}).reset_i
df_directors=df_directors[df_directors['Director']!='Unknown Director']
plt.figure(figsize=(15,8))
plt.barh(df_directors[::-1]['Director'], df_directors[::-1]['title'],color=['indigo
plt.xlabel('Number of Movies')
plt.ylabel('Popular Directors')
plt.show()
```



Popular directors across movies in India:-

'Rajiv Chilaka',

'Suhas Kadav',

'David Dhawan',

'Umesh Mehra',

'Anurag Kashyap',

'Ram Gopal Varma',

'Dibakar Banerjee',

'Zoya Akhtar',

'Tilak Shetty',

'Rajkumar Santoshi',

'Priyadarshan',

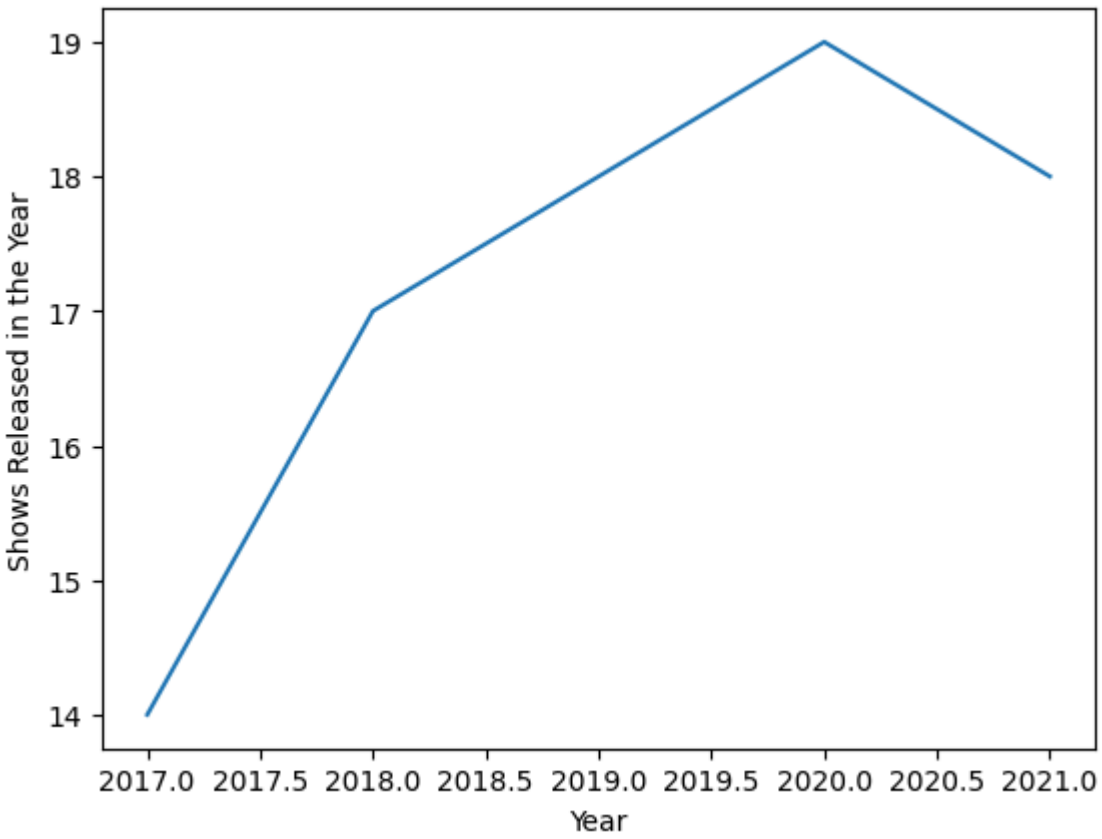
'Sooraj R. Barjatya',

'Ashutosh Gowariker',

'Milan Luthria'

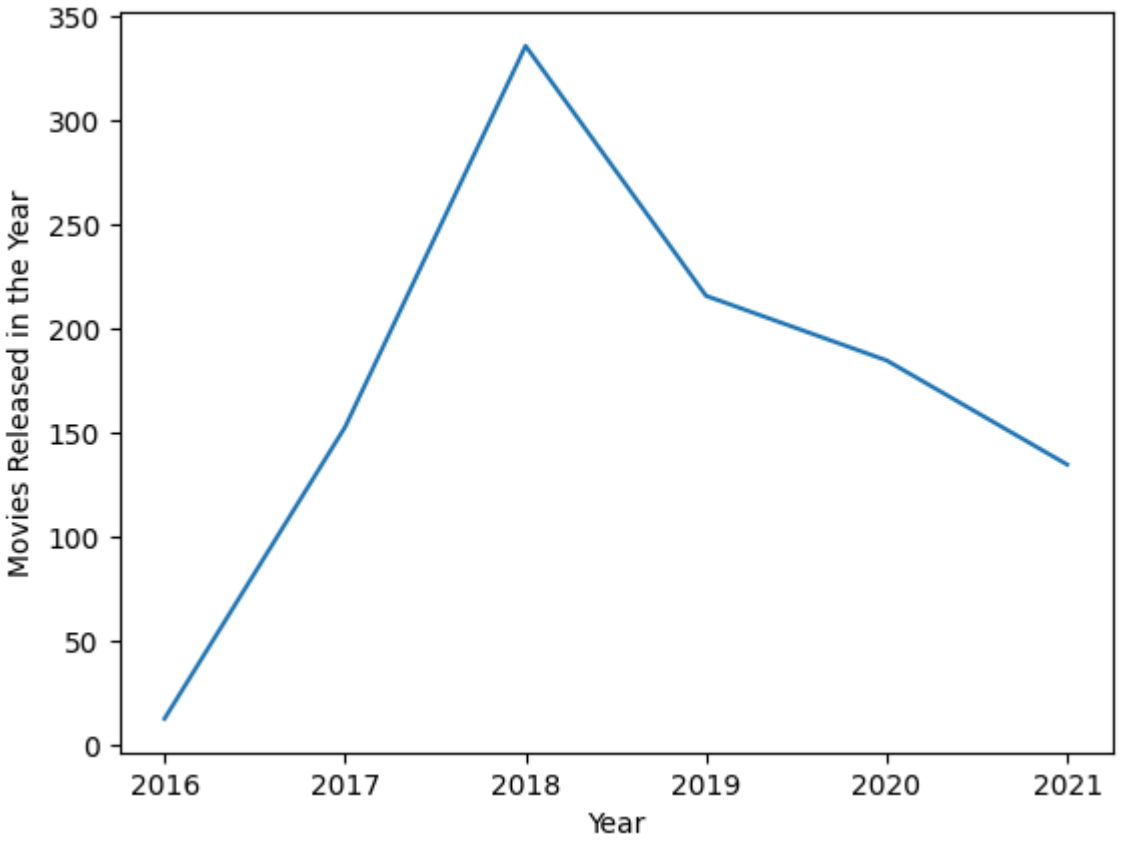
In [187…

```
df_year=df_india_shows.groupby(['year']).agg({"title":"nunique"}).reset_index()
sns.lineplot(data=df_year, x='year', y='title')
plt.ylabel("Shows Released in the Year")
plt.xlabel("Year")
plt.show()
```



In [188…

```
df_year=df_india_movies.groupby(['year']).agg({"title":"nunique"}).reset_index()
sns.lineplot(data=df_year, x='year', y='title')
plt.ylabel("Movies Released in the Year")
plt.xlabel("Year")
plt.show()
```

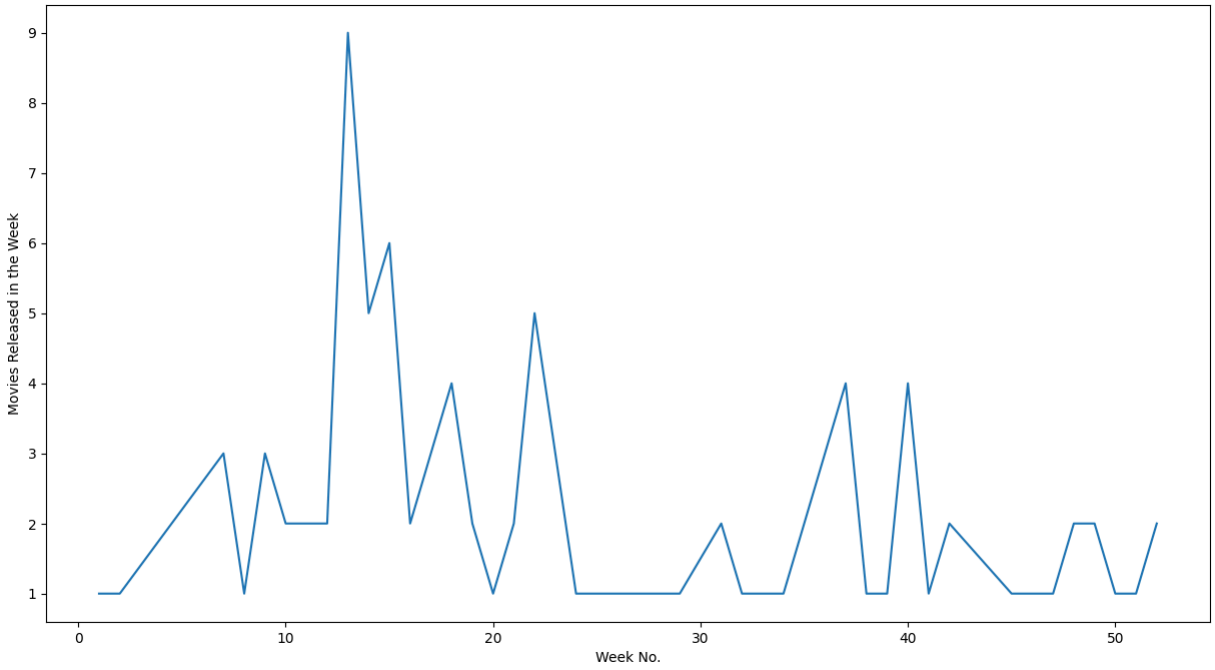


In India,TV Shows were increasingly being added till 2020, though the addition of shows reduced in 2021.

In India, Movies were increasingly added till 2018 but it has been a huge downhill since then. Now that's preposterous, since and soemthing has to be recommended to the Netflix Team with regards to that.

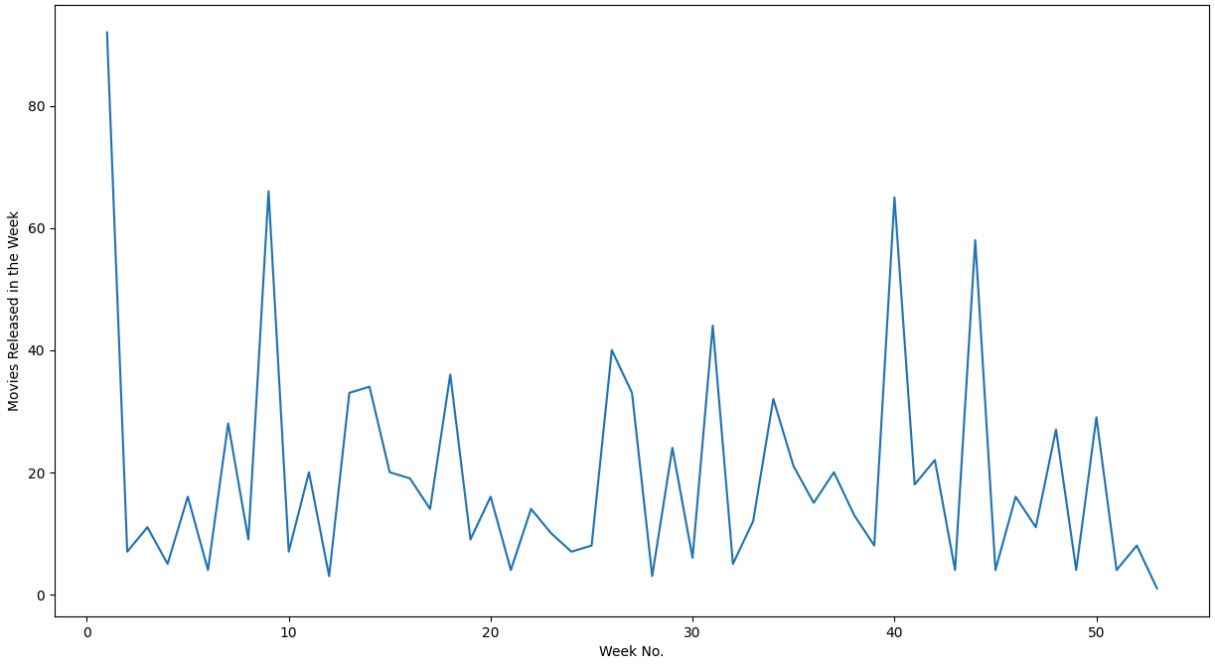
In [189…

```
df_week=df_india_shows.groupby(['week']).agg({"title":"nunique"}).reset_index()
plt.figure(figsize=(15,8))
sns.lineplot(data=df_week, x='week', y='title')
plt.ylabel("Movies Released in the Week")
plt.xlabel("Week No.")
plt.show()
```



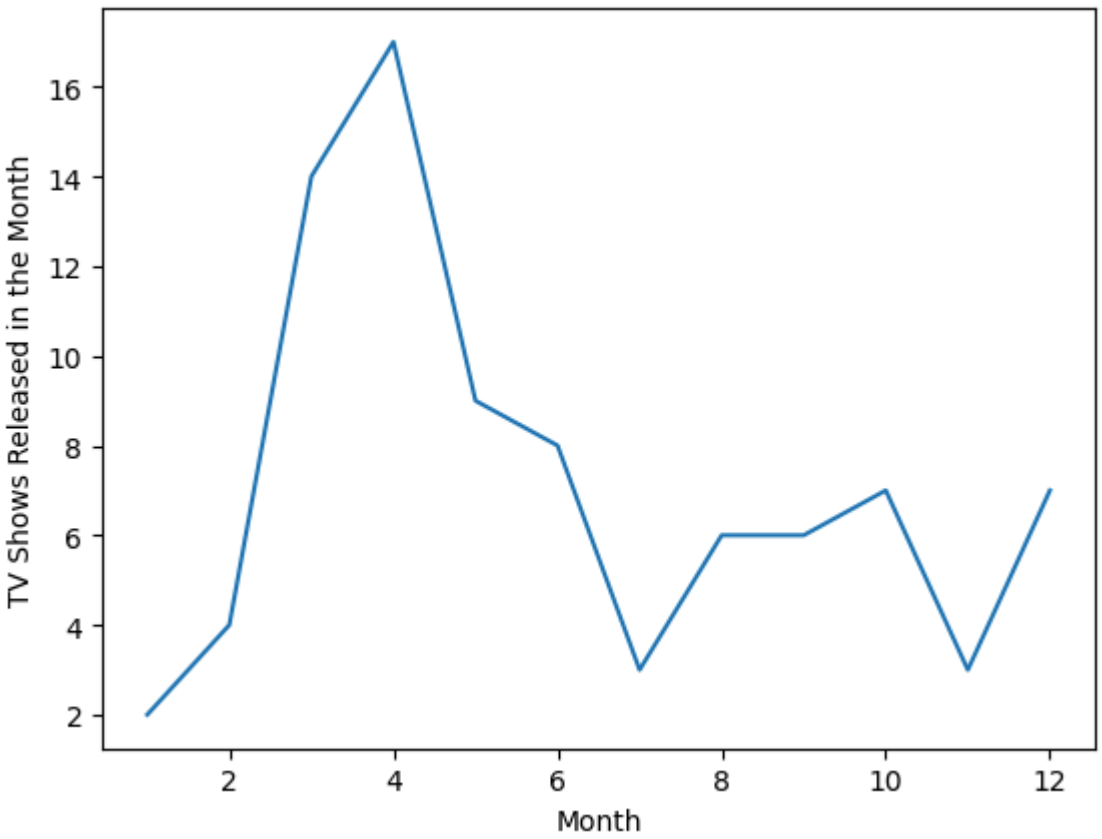
In [190…

```
df_week=df_india_movies.groupby(['week']).agg({"title":"nunique"}).reset_index()
plt.figure(figsize=(15,8))
sns.lineplot(data=df_week, x='week', y='title')
plt.ylabel("Movies Released in the Week")
plt.xlabel("Week No.")
plt.show()
```



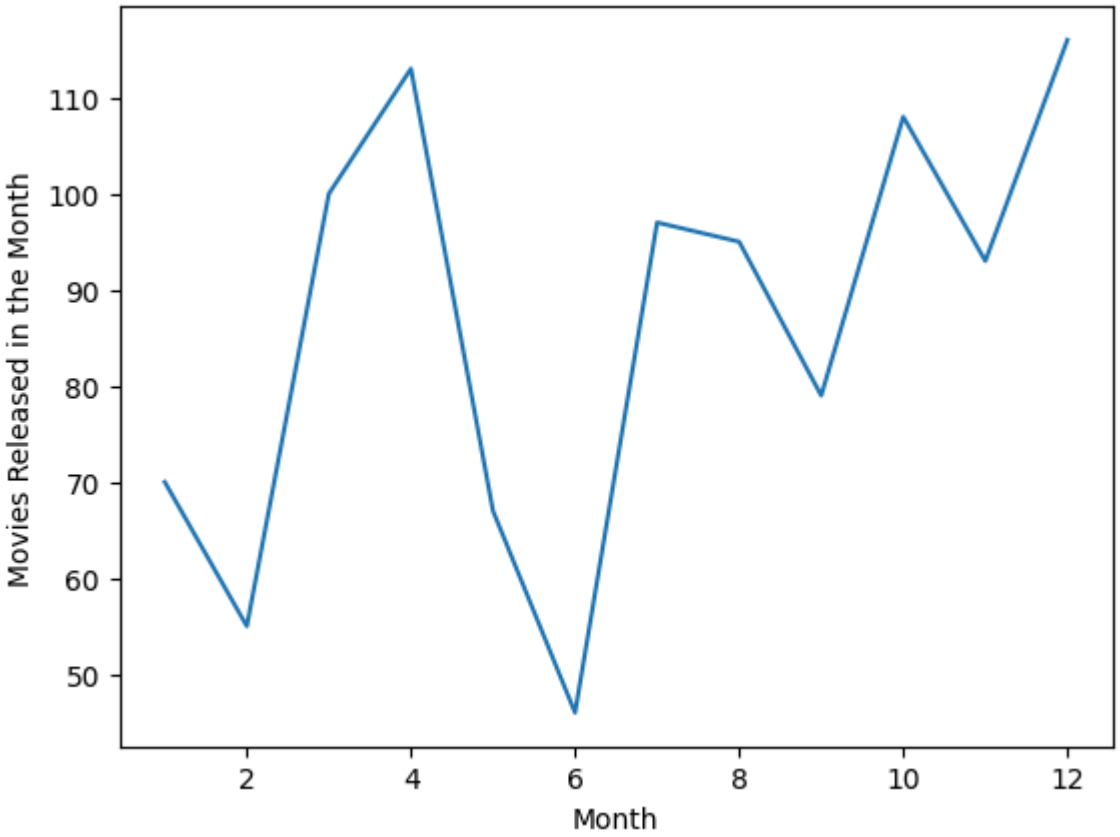
In [191…

```
df_month=df_india_shows.groupby(['month']).agg({"title":"nunique"}).reset_index()
sns.lineplot(data=df_month, x='month', y='title')
plt.ylabel("TV Shows Released in the Month")
plt.xlabel("Month")
plt.show()
```



In [192…

```
df_month=df_india_movies.groupby(['month']).agg({"title":"nunique"}).reset_index()
sns.lineplot(data=df_month, x='month', y='title')
plt.ylabel("Movies Released in the Month")
plt.xlabel("Month")
plt.show()
```

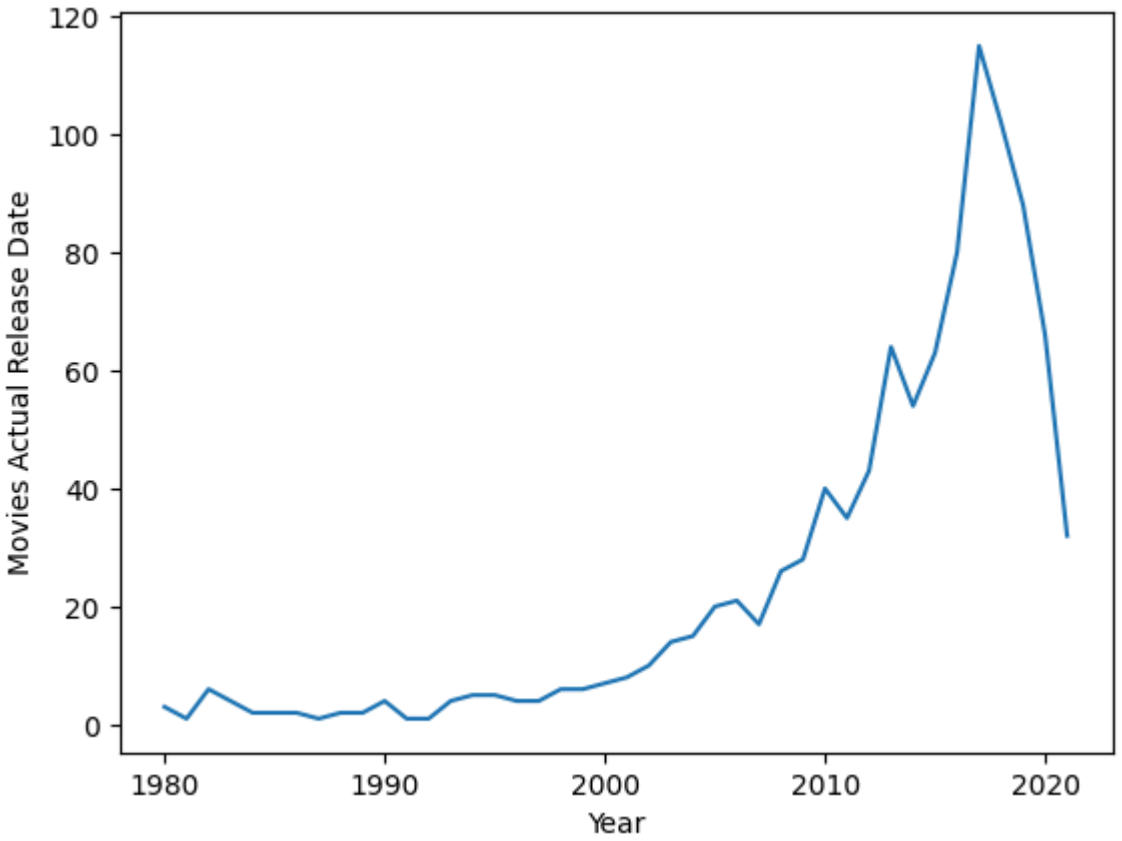


TV Shows are added in Netflix by a tremendous amount in April in India

Movies are added in Netflix in India by a tremendous amount in first week/last month of current year and first month of next year

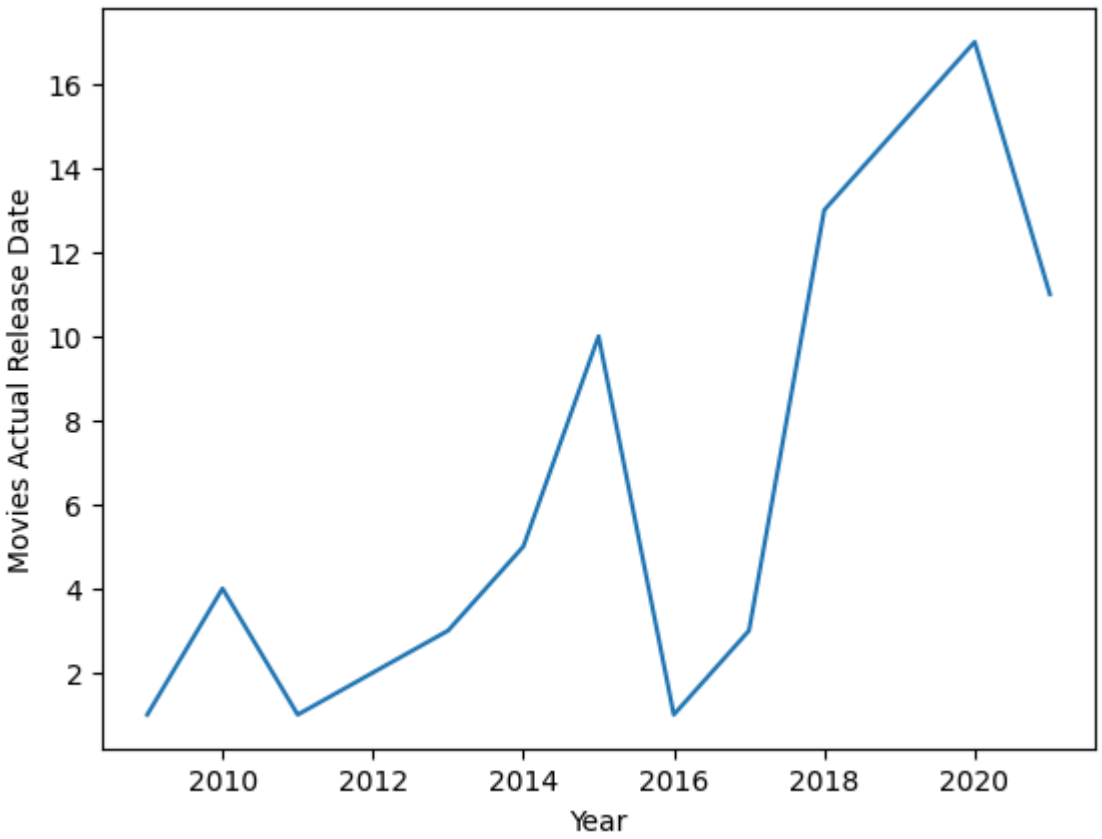
In [193…

```
df_release_year=df_india_movies[df_india_movies['release_year']>=1980].groupby(['re
sns.lineplot(data=df_release_year, x='release_year', y='title')
plt.ylabel("Movies Actual Release Date")
plt.xlabel("Year")
plt.show()
```



In [194…

```
df_release_year=df_india_shows[df_india_shows['release_year']>=1980].groupby(['rele
sns.lineplot(data=df_release_year, x='release_year', y='title')
plt.ylabel("Movies Actual Release Date")
plt.xlabel("Year")
plt.show()
```



The understandable trend amongs movies and TV Shows across India in Netflix is the reduction of movies after 2020

In [195…

```
#Analysing a combination of actors and directors
df_india_movies['Actor_Director_Combination'] = df_india_movies.Actors.str.cat(df_i
df_india_movies_subset=df_india_movies[df_india_movies['Actors']!='Unknown Actor']
df_india_movies_subset=df_india_movies_subset[df_india_movies_subset['Director']!='
df_india_movies_subset.head()
```

```
C:\Users\ARJUN\AppData\Local\Temp\ipykernel_20096\795884279.py:2: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  df_india_movies['Actor_Director_Combination'] = df_india_movies.Actors.str.cat(df_
india_movies.Director, sep=' and ')
```

Out[195...

	title	Actors	Director	Genre	country	show_id	type	date
621	Avvai Shanmughi	Kamal Hassan	K.S. Ravikumar	Comedies	India	s23	Movie	Se
622	Avvai Shanmughi	Kamal Hassan	K.S. Ravikumar	International Movies	India	s23	Movie	Se
629	Avvai Shanmughi	Nassar	K.S. Ravikumar	Comedies	India	s23	Movie	Se
630	Avvai Shanmughi	Nassar	K.S. Ravikumar	International Movies	India	s23	Movie	Se
631	Avvai Shanmughi	S.P. Balasubrahmanyam	K.S. Ravikumar	Comedies	India	s23	Movie	Se

In [196...

```
df_india_shows['Actor_Director_Combination'] = df_india_shows.Actors.str.cat(df_ind
df_india_shows_subset=df_india_shows[df_india_shows['Actors']!='Unknown Actor']
df_india_shows_subset=df_india_shows_subset[df_india_shows_subset['Director']!='Unk
df_india_shows_subset.head()
```

```
C:\Users\ARJUN\AppData\Local\Temp\ipykernel_20096\3189041927.py:1: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  df_india_shows['Actor_Director_Combination'] = df_india_shows.Actors.str.cat(df_in
dia_shows.Director, sep=' and ')
```

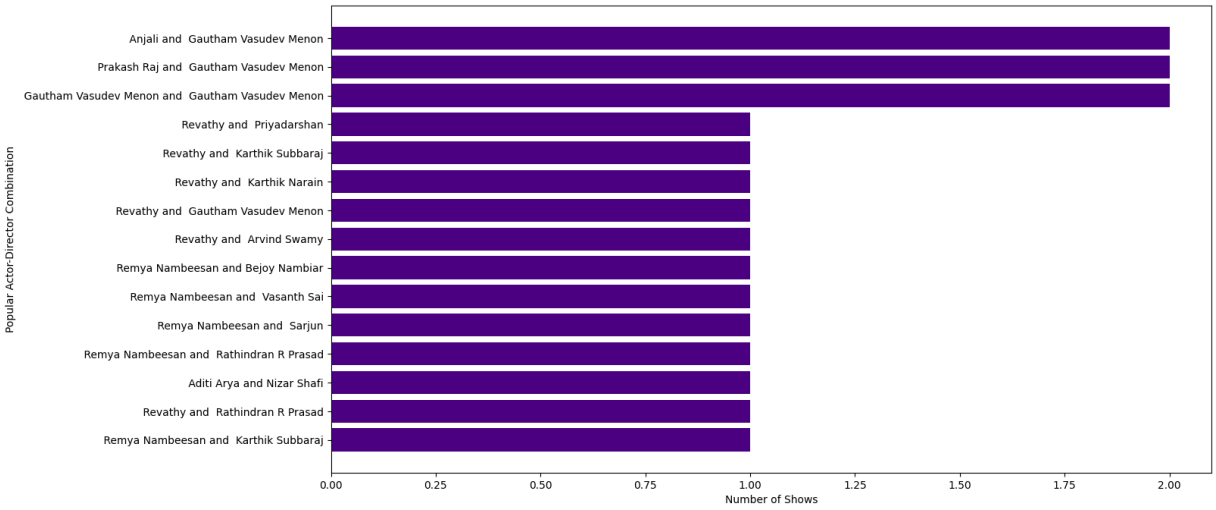
Out[196...

	title	Actors	Director	Genre	country	show_id	type	date_added	release_
7005	Navarasa	Suriya	Bejoy Nambiar	TV Shows	India	s298	TV Show	August 6, 2021	
7006	Navarasa	Suriya	Priyadarshan	TV Shows	India	s298	TV Show	August 6, 2021	
7007	Navarasa	Suriya	Karthik Narain	TV Shows	India	s298	TV Show	August 6, 2021	
7008	Navarasa	Suriya	Vasanth Sai	TV Shows	India	s298	TV Show	August 6, 2021	
7009	Navarasa	Suriya	Karthik Subbaraj	TV Shows	India	s298	TV Show	August 6, 2021	

In [197...

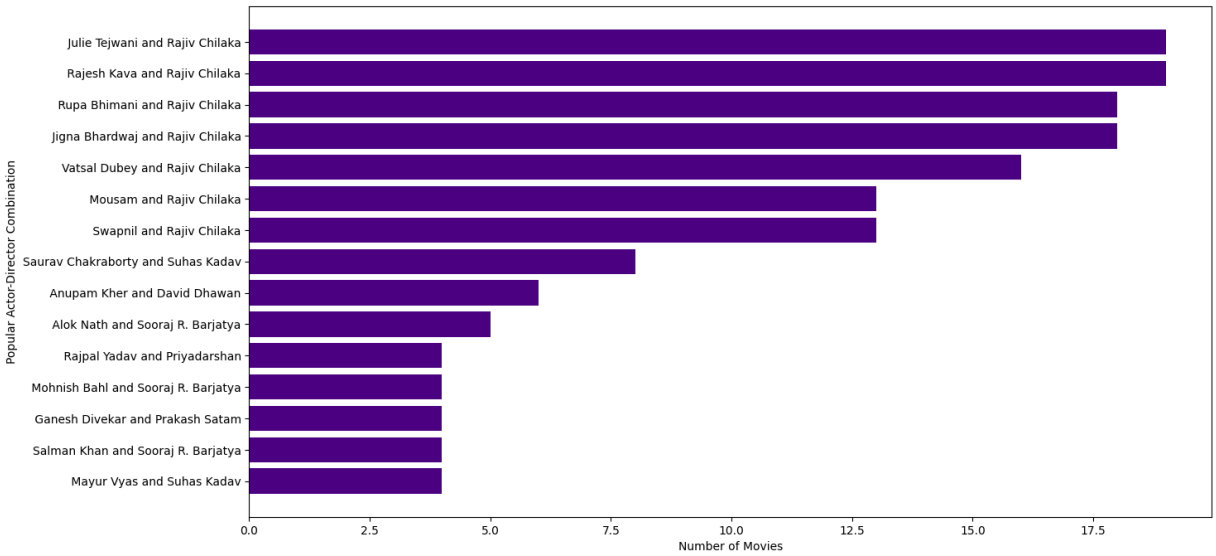
```
df_actors_directors=df_india_shows_subset.groupby(['Actor_Director_Combination']).a
plt.figure(figsize=(15,8))
plt.barh(df_actors_directors[::-1]['Actor_Director_Combination'], df_actors_directo
```

```
plt.xlabel('Number of Shows')
plt.ylabel('Popular Actor-Director Combination')
plt.show()
```



In [198…

```
df_actors_directors=df_india_movies_subset.groupby(['Actor_Director_Combination']).
plt.figure(figsize=(15,8))
plt.barh(df_actors_directors[::-1]['Actor_Director_Combination'], df_actors_directo
plt.xlabel('Number of Movies')
plt.ylabel('Popular Actor-Director Combination')
plt.show()
```



In [199…

```
df_india_movies[df_india_movies['Director']=='Rajiv Chilaka'].head(3)
```

Out[199...

	title	Actors	Director	Genre	country	show_id	type	date_added	release_
10058	Chhota Bheem - Neeli Pahaadi	Vatsal Dubey	Rajiv Chilaka	Children & Family Movies	India	s407	Movie	July 22, 2021	
10059	Chhota Bheem - Neeli Pahaadi	Julie Tejwani	Rajiv Chilaka	Children & Family Movies	India	s407	Movie	July 22, 2021	
10060	Chhota Bheem - Neeli Pahaadi	Rupa Bhimani	Rajiv Chilaka	Children & Family Movies	India	s407	Movie	July 22, 2021	

It seems that Rajiv Chilaka has worked on Chota Bheem and has been able to create some good content in its movies. He can be relied on for more Chota Bheem stories

The Most Popular Actor Director Combination in Movies Across India are:-
'Rajesh Kava and Rajiv Chilaka',
'Julie Tejwani and Rajiv Chilaka',
'Rupa Bhimani and Rajiv Chilaka',
'Jigna Bhardwaj and Rajiv Chilaka',
'Vatsal Dubey and Rajiv Chilaka',
'Mousam and Rajiv Chilaka',
'Swapnil and Rajiv Chilaka',
'Saurav Chakraborty and Suhas Kadav',
'Smita Malhotra and Tilak Shetty',
'Anupam Kher and David Dhawan',
'Salman Khan and Sooraj R. Barjatya',

Recommendations

1. The most popular Genres across the countries and in both TV Shows and Movies are Drama, Comedy and International TV Shows/Movies, so content aligning to that is recommended.

2)Add TV Shows in July/August and Movies in last week of the year/first month of the next year.

3)For USA audience 80-120 mins is the recommended length for movies and Kids TV Shows are also popular along with the genres in first point, hence recommended.

4)The target audience in USA and India is recommended to be 14+ and above ratings

5)Add movies for Indian Audience, it has been declining since 2018.

6. While creating content, take into consideration the popular actors/directors for that country. Also take into account the director-actor combination which is highly recommended.

In [ ]: